I/O NEWS

PerSci Q & A
Curses
Winning at Copyrights

THE OFFICIAL PUBLICATION OF THE INTERNATIONAL ASSOCIATION OF CROMEMCO USERS

Volume Five, Number Six

Single Copy Price $10.00

Cromemco's System 220
The Ace of Supermicros

by Bill Jaenicke

Moving briskly through the dawn's early light, a U.S. Air Force fighter pilot heads towards his F-16 Falcon multi-role aircraft. Besides his skill and know-how, he carries with him a DTC/LR (Data Transfer Cartridge/Loader Reader).

Once aboard the fighter jet he begins his pre-flight check. He plugs in the DTC, whose memory contains details on the mission flight plan and data for initializing his on- board instrumentation. It sets the radio frequencies for ground control at the base and for the in-flight air control connect points, as well as calibrating the altimeter and related instruments to the current barometric pressure. These and other pre-flight procedures, which used to delay take-off for as long as an hour (burning fuel all the while), are completed in a matter of minutes.

The Mission Planning System, which the pilot used to program the DTC, has also provided the pilot with the safest and most effective flight plan. This information is also loaded into the 16's on-board systems and will be available to the pilot throughout the course of his mission. He takes off, secure in the knowledge that this new technology, combined with his own special skills, will give him an edge in successfully completing his mission.

Mission Planning System (MPS)

The scenario described above is not science fiction. The Mission Planning System is real, and at its heart is Cromemco's System 220 — serving as data ground terminals for F-16's deployed throughout the world.

The System 220 uses Cromemco's XXU board with 68020/68881

Continued on page 30

Micro Systems for Low-Earth-Orbit Satellite Tracking & Data Acquisition

by Robert J. Diersing

Editor's Note: *Words in* **boldface** *are included in the glossary of terms at the end of this article.*

ABSTRACT

This article describes two microcomputer systems used for low-earth-orbit satellite tracking and data acquisition. The systems were developed for use with spacecraft operated in the Amateur Satellite Service but could be adapted for use with other satellites. After some background information on the Amateur Satellite Service and the spacecraft currently in operation, the details of each system are presented.

Continued on page 10

ProCall/PC+
PC-to-Cromix Communication and Beyond

by Bill Jaenicke

Title: ProCall/PC+ — PC-to-Cromix Communication and Beyond
Copyright Bill Jaenicke 1987

As Cromemco users, many of us find ourselves caught in a tug-of-war. On the one side is our Cromix system—a system we've come to respect and depend on for its power, expandability and elegance. On the other side is the allure of the PC, with its easy affordability and vast reservoir of inexpensive software. Some have already compromised, as others surely will, and find themselves with one or more PCs and a Cromix system. The good news is that

Continued on page 28

ONLY ONE COMMUNICATIONS PROGRAM HAS BEEN CALLED EASIER TO USE THAN CROSSTALK, SUPPORTS RFILE, SFILE, 3102 TERMINAL EMULATION, AND A FILE TRANSFER PROTOCOL THAT'S NAMED AFTER A FROG.

ProCall/PC

Yep, it's true. Only one program has what it takes to be set apart from the rest. Sure, we put in features like Xmodem, DEC VT-100 and VT-52 emulation, but we didn't stop there. If you're looking for a powerful communications program to help you share information with public information services, BBS systems, and your Cromemco systems, the answer is staring you right in the face.

Now all versions of ProCall, and ProCall/PC *plus* support the Kermit file transfer protocol, and we've made several other changes that won't go unnoticed either.

Ask your dealer about ProCall and ProCall/PC. Or ask us:

ProtoMatrix Software Development
1145 Park Heights Drive
Milpitas, CA 95035 USA (408) 263-8665

PSD

table of contents

March/April, 1987 Volume Five, Number Six

I/O NEWS

The Official Publication of The International Association of Cromemco Users is available through membership in the association. Editorial and advertising policies are designed for the enlightenment of the members in regard to new uses for, and developments of, Cromemco products and other products compatible with Cromemco systems.

I/O NEWS [ISSN 0274-9998] is published bi-monthly by The International Association of Cromemco Users (a California corporation), 24843 Del Prado, Suite 473, Dana Point, CA 92629-2852. General Offices are at 34021 Granada #B, Dana Point, CA 92629. Telephone: (714) 661-9764. Second Class Postage Paid at Dana Point and Santa Ana, CA.

POSTMASTER: Send address changes to I/O NEWS, 24843 Del Prado, Suite 473, Dana Point, CA 92629-2852.

Subscriptions to I/O NEWS are entered with membership in The IACU. Yearly memberships may be purchased for $42 (U.S. delivery address), $49 (delivery address in Canada or Mexico), and $60 (other international delivery addresses). Contact IACU for multi-year membership rates. Back issues of I/O NEWS are available for $10.00 per issue. Please note: all prices are in U.S. dollars. Return postage must be included with all manuscripts and photos submitted if they are to be returned. The IACU and I/O NEWS accept no responsibility for the return of unsolicited materials. All rights in letters sent to IACU and I/O NEWS will be treated as unconditionally assigned for publication and copyright to comment editorially and to edit.

William E. Jaenicke
Editor and Publisher

Lisa B. Jaenicke
Business Manager

Art Direction: Howard Millman
Printing: Hallmark Litho
Typography: Western Outdoors

COVER FEATURES

ARTICLES & FEATURES

DEPARTMENTS

U.S. Postal Service
STATEMENT OF OWNERSHIP, MANAGEMENT AND CIRCULATION

1A. Title of Publication: I/O NEWS
1B. Publication No.: 0 2 7 4 9 9 9 8
2. Date of Filing: 9/30/85
3. Frequency of Issue: Bi-monthly
3A. No. of Issues Published Annually: 6
3B. Annual Subscription Price: $42.-U.S. $49.-CN/MX $60. Int'l
4. Complete Mailing Address of Known Office of Publication: 34021 Granada, Suite B, Dana Point, CA 92629
ALL MAIL TO: P.O. Box 17658, Irvine, CA 92713
5. Complete Mailing Address of the Headquarters of General Business Offices of the Publisher: Same as item 4.
6. Full Names and Complete Mailing Address of Publisher, Editor, and Managing Editor
Publisher: William E. Jaenicke (same address as item 4.)
Editor: William E. Jaenicke " " " " " "
Managing Editor: William E. Jaenicke " " " " " "
7. Owner

Full Name	Complete Mailing Address
The International Association of Cromemco Users, Inc.	same as item 4.
Harry T. Garland, Ph.D.	280 Bernardo Ave., Mt. View, CA 94039

8. Known Bondholders, Mortgagees, and Other Security Holders Owning or Holding 1 Percent or More of Total Amount of Bonds, Mortgages or Other Securities

Full Name	Complete Mailing Address
NONE	

9. For Completion by Nonprofit Organizations Authorized to Mail at Special Rates
(1) Has not changed during preceding 12 months (2) Has changed during preceding 12 months

10. Extent and Nature of Circulation	Average No. Copies Each Issue During Preceding 12 Months	Actual No. Copies of Single Issue Published Nearest to Filing Date
A. Total No. Copies (Net Press Run)	3,179	2,500
B. Paid and/or Requested Circulation 1. Sales through dealers and carriers, street vendors and counter sales	750	505
2. Mail Subscription (Paid and/or requested)	1,510	1,305
C. Total Paid and/or Requested Circulation (Sum of 10B1 and 10B2)	2,260	1,810
D. Free Distribution by Mail, Carrier or Other Means Samples, Complimentary, and Other Free Copies	66	68
E. Total Distribution (Sum of C and D)	2,326	1,878
F. Copies Not Distributed 1. Office use, left over, unaccounted, spoiled after printing	853	622
2. Return from News Agents	-0-	-0-
G. Total (Sum of E, F1 and 2 - should equal net press run shown in A)	3,179	2,500

11. I certify that the statements made by me above are correct and complete. Signature and Title of Editor, Publisher, Business Manager, or Owner

PS Form 3526, July 1984

INPUT...

Editor:

Dr. Stephen Huber's benchmark article in *I/O NEWS* Volume V, No. 5 was well written, concise, comprehensive, and a service to the Cromemco user community. I read it three times. Since my DPU system has neither the MAXIMIZER nor FORTRAN '77, I cannot contribute more benchmark data. However, I can pass along a tip that could solve two difficulties that he encountered. Dr. Huber lamented the expense of the faster KZ memory and 10Mhz DPU, and reported disk I/O difficulties with an 8Mhz DPU modified for 10MHz. The answer is Kurt, who works for Excalibur. Kurt upgraded my 256KZ last year, completely fixing a recurring problem which I had experienced with it. Since I was so happy with the quality and price of that upgrade, I asked him to boost my old DPU to 10Mhz and upgrade by 16FDC while he was at it. The memory upgrade cost a reasonable amount, and I'm still astonished at what a bargain the processor speedup was. Of course, it has run perfectly for two months now and works beautifully with the new STDX.

Speaking of the STDX (the 10MHz STDC), pricey as it is, it is the Winnie controller of choice. I just installed it with an 85Mb Miniscribe 6085 (28ms) from Direct-To-You (very competitive prices, fast service: (408) 437-1001). This is the entry for that drive to be placed in Richard Quinn's hard disk parameters table from Tec Tips, Vol. V, No. 3:

Manufacturer: Miniscribe
Model: 6085
Capacity: 85 Mbyte
Cylinders: 1024
Heads: 8
Precomp: 1024

Thanks Dr. Huber, thanks Kurt, and thank you, Richard, for those init* tips.

Joseph L. Brothers
Warecraft
501 North 36th St. –138
Seattle, WA 98103
(206)524-5403

Editor:

My letter is prompted by your comments in the "INPUT" column of the November/December issue of *I/O NEWS* about non-compatibility with IBM. I agree with your thesis that Cromemco has never seen itself in head-to-head competition with IBM. But your response fails to take into consideration some realities which need serious evaluation, especially by Cromemco's development staff.

For those of us who started out in the CP/M world of the late 70's, Cromemco made it easy to move into multiuser computing. The CDOS simulator permitted us to bring into the CROMIX world many of the tools we had been running under CP/M. Even if the software did not run perfectly under CDOS, a little hacking with DEBUG would usually fix the problem sufficiently.

But Cromemco has never provided a similar door of opportunity for those coming out of the MS-DOS world. The simple presence of an MS-DOS emulator, similar to the CDOS simulator, would have permitted those who entered computing in the post-PC-DOS era to make the same type of transition to Cromemco multiuser machines that was possible for those of us with the CP/M backgrounds.

In my own circle of friends are several professionals who would have bought System 400s had they not been compelled to force an entirely new learning curve on employees who had become dependent on certain MS-DOS products. What these professionals wanted was the added flexibility, security, and power of a UNIX-like environment without having to abandon the MS-DOS product world entirely.

Microsoft, sensing that such markets are growing, plans a version of XENIX for 80286 machines that will permit individual users to run MS-DOS applications in a shell within XENIX. In my own judgement, that will prove an extremely attractive OS — one I would find hard to resist myself.

Not only new clients, but existing ones are being lost by Cromemco because of the absence of MS-DOS capabilities. I regularly help non-profit organizations set up in-house computing by buying or soliciting the contribution of used Cromemco equipment. As I respond to ads for used Cromemco machines, I find that fully four out of every five ads are from individuals who are moving away from Cromemco. Their number one reason is the inability to tap into the affordable, wide-ranging options of MS-DOS software.

Every time I talk to people who have made that choice, I find myself tempted to join them. On a finite budget, I simply cannot afford to pay the cost of UNIX applications — such as table top publishing — when everything I need to do is easily accomplished at a fraction of the price on software available for MS-DOS.

Let me hasten to add that my commitment to Cromemco is neither newborn nor nominal. I placed my order for a C-10 before actual manufacture had begun. Later I bought one of the earliest DPU machines and more recently an XPU version of the System 100. During the early '80's, as president of a small liberal arts college, I worked closely with Dr. Garland in creating the first campus-wide computer literacy program in the Northwest using dozens of C-10s and C-5s with two System Threes and a Z-2h as multiuser hosts.

Even with that commitment, however, I am regularly frustrated when working on joint development projects with people who are themselves using MS-DOS tools. Over the past year, in my naval reserve work, I have been producing a suite of analytical programs for managing a national recruiting effort. The Navy will be running those programs on MS-DOS computers making use of standard IBM-type software. I am sorely limited in the amount of development work I can do on my own equipment, however, without some type of bridge from the CROMIX/UNIX world to MS-DOS. I doubt that I'm the only Cromemco devotee who regularly faces that kind of problem.

In concert with Rich Quinn, I am currently evaluating some prototype modules of what may eventually become a transparent MS-DOS emulator

for Cromix. If the entire package works as well as the modules in hand, we may have the answer to the concerns above. But if that effort falls short, I hope someone else will find a way to fashion a Cromemco-to-MS-DOS bridge. And I would hope that Cromemco could see its way clear to put a blessing on that effort as a means of securing new markets and holding onto some old ones.

Fraternally,

Michael C. Armour
6401 Mercedes Avenue
Dallas, TX 75214

Editor:

I disagree with you when you say that it is a futile exercise to compare an IBM PC with a Cromemco system running UNIX or a look-alike. It is certainly true from a technical point of view that MS-DOS is much more arcane than Cromix. The point here is, I believe, the large amount of quality software available on MS- or PC-DOS (you make the argument with CP/M in your reply). The facts are that many computer makers have some kind of MS-DOS emulation available: DEC has the VAXmate, a diskless AT to run with any VAX; SUN and Apollo have emulation capabilities in their workstations; and even Apple is talking of doing it on the new Mac. I know many other people who want it, and I believe Cromemco should be doing what their customers ask for.

By the way, if Cromemco decided it was not worth the effort (there are also some good reasons for this, e.g., the price of XT clones), would Cromemco be willing to disclose details on their new coprocessor interface: z80.bin and the zio driver. There are a number of S-100 8086 boards on the market that could be easily modified to look like an IOP-X to the system.

I also have a suggestion with respect to the feedback to Cromemco. If someone feels the system should be enhanced in new ways, he could file, presumably with IACU, a so called System Improvement Request, a SIR. The SIR should explain what is the missing functionality and why it makes life hard, and make a proposal of the new commands, etc... to solve the perceived problem (the technical solution is of course Cromemco's business).

After a while the SIR's could be published, and a poll would be taken among the users who would vote for the most useful proposals. The poll result would then be submitted to Cromemco for comments.

Thank you for keeping I/O NEWS interesting.

Sincerely Yours,

Alberic Muller
Rue du Jura 12
CH-2525 Le Landeron
Switzerland

Editor's Reply:

As evidenced by the number and intensity of the replies that my earlier statements have generated, it is safe to say that the issue has not been laid to rest. Apparently, complete satisfaction is required.

Nevertheless, Cromemco-to-MS-DOS bridges are available, and have been so for some time now. I refer to programs for copying various disk formats, development software for which there are versions that run under Cromix and Unix and MS-DOS (e.g., Informix), and communications programs which allow for the interconnection of MS-DOS machines and Cromemco machines for the purpose of file transfers or distributed processing, ProCall/PC for example.

Admittedly, the bridges are a little narrow in that they don't enable the Cromemco user to load MS-DOS programs into his Cromemco memory and execute them directly on the Cromemco machine. Even if there was such an emulator to allow for this, MS-DOS programs which make use of the graphics capabilities of the bit-mapped PC terminal would not run. That would remove a big slice from the MS-DOS software pie, and lead to less than complete satisfaction.

Full compatibility would require additional hardware in terms of graphics cards and monitors. It can be done—much (but not all) of the required hardware and software components are presently available—it's that little bit that's missing, and then bringing it all together, that is holding things up. That's because it's going to cost money to design that extra bit and write that special driver.

But that is what free enterprise is about: supply and demand. If there's enough demand you know there are plenty of companies that could provide for that demand and turn a profit at the same time.

The problem facing those of us who would like to be able to run MS-DOS programs directly on our Cromemco systems is one of making our desires known. It's one thing to approach a vendor and say, "I've talked to a lot of people that would be willing to buy Product X if it existed.." and quite another to approach the same vendor with written testimonials from so many individuals stating their interest in product X. So if you really want to see it happen write a letter to us stating so. Then we just might be able to do something about it.

In the meantime let's take a look at where we are now: inexpensive PC clones for running inexpensive PC software coupled with suitable communications software that enables the PCs to act as terminals to more sophisticated and secure Cromix and Unix systems — on call when their multi-user capabilities and superior hardware are needed. From my viewpoint it's the next best thing to having your pie and eating it too.

In regards to the notion of a SIR survey, it sounds like a good idea. We'll work on a suitable format and publish it in the next issue.

Bill

OUTPUT...

New Mailing Address

The mailing address for *I/O NEWS* and the I.A.C.U. has been changed. Please take the time now to update your records accordingly:

The I.A.C.U and I/O NEWS
24843 Del Prado, Suite 473
Dana Point, CA 92629-2852

Although the old mailing address will still be active for another year, we will receive your mail much sooner if it is addressed as shown above.

That aside, let's take a look at this issue. For one, we've taken a slight departure in format and content. Some of the regularly appearing columns such as TEC TIPS, 32K CLASSROOM, INSIDE CROMIX, NEW PRODUCTS, and BITS & BYTES aren't appearing this time. The reason is that we have received a number of noteworthy editorial contributions from our members, and there just wasn't enough space to run all of it. So we gave those column editors a well-deserved rest, and have presented the articles which we think you will find most interesting.

As such, this edition includes some of the more "high-tech" applications that Cromemco computers are being utilized for. At the top is the Mission Planning System which is described in the front cover article, "Cromemco's System 220: The Ace of Supermicros."

Lisa Jaenicke

This system truely expands the bounds of supermicro capability, incorporating the latest XXU technology. Then there is Robert J. Diersing's contribution, "Micro Systems for Low-Earth-Orbit Satellite Tracking and Data Acquisition." We hope that you'll find it as interesting and thought provoking as we did.

In response to all the hoopla regarding PC clones, we've put together a review of the premier modem communications system for PCs: ProCall/PC Plus. As you'll see, this system may provide the answer to a problem facing many of us: how to integrate the PCs and Cromix.

On a more mundane level, we offer the first of a series of articles which explain how to go about copyrighting software. As many of you develop commercial applications, this should be of particular interest. The articles are written by Paul Hentzel, who has been Cromemco's copyright, trademark, and patent attorney since they started business.

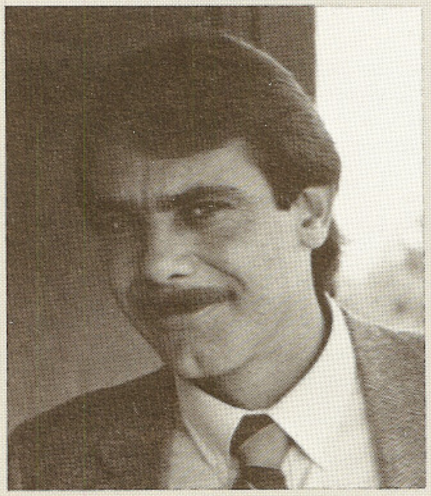
As we have received many inquiries regarding the service and maintenance of Persci 299B disk drives, we went to the authority on the subject, John Bush of Peripheral Labs, and asked him to comment on the subject. The result is his "PerSci Q & A" article.

We've also presented a substantial number of USER NOTES, all submitted by IACU members, which range from tips for using the C-10 to tips for using UNIX. And then there's CURSES— but you'll have to read Soup Campbell's article to find out what that is all about.

This issue also heralds the return of SOFT TIPS, with its new editor, Bob Brown of Excalibur Computers. Although this installment offers some useful C routines for formatted input, future installments will be geared more towards end-users of commercially available applications software.

All in all, this is an information-packed issue. So get to it! And as usual, your suggestions and comments are always welcome (but be sure to send them to our new mailing address).

Bill Jaenicke
Editor



Bill Jaenicke

PerSci Q&A

by John A. Bush

In the past, as Lead Technician at PerSci, Inc., and more recently with Peripheral Labs, I've fielded many a question about the PerSci 299B Floppy Disk Drive. This article is in response to the many thousands of Cromemco owners who have expressed interest in gaining more information about their PerSci disk drives. I will also provide some interesting historical notes and attempt to dispel some of the myths associated with the PerSci disk drive.

The following are my responses to the most common questions put to me.

Q1. Are PerSci drives economical to repair?

A1. Yes and no.

In 1979, the year the PerSci 299B was announced, *Computer Design Magazine* voted the PerSci disk drive the Product of the Year. Orders for the then prototype were very heavy. PerSci management of the time was under extreme pressure to fill these orders, even though they lacked the facilities to manufacture in the necessary quantity. One corner cutting plan involved the first 1500 299B's being QA'ed by the engineering department. Many of the drives failing the final test phase were the subject of "custom" engineering just to get them out of the door. None of these "custom" changes were ever documented and the resulting products were marginal at best. Therefore this first generation, serial numbers below D2500, can be extremely difficult and expensive to support. If you have one of these drives and it has been giving you a lot of trouble, I suggest you replace it! The drive serial number can be found at the rear of the drive below the large circular spindle motor. It is the bottom number on the approximately one-inch square, silver metallic tag.

Another difficulty associated with servicing the PerSci disk drive is related to the 180 engineering changes that were accomplished on the drive between serial numbers D2500 and D7000. Only eight levels of documentation were developed. As a result, the highly skilled technician—well equipped and well meaning—may encounter difficulties as supporting documentation doesn't exist. Manuals from PerSci and Cromemco for these machines include reprints of early generation schematics and are of limited use in troubleshooting. Our hands-on experience coupled with a complete set of PerSci documentation (extremely rare) allows us the exclusive ability to accomplish virtually any repair on any generation of the PerSci disk drive.

Another very common practice among persons and organizations who perform service on the drive is "light-screwing" the read/write heads. To complete the manual head alignment the retaining screws must be securely tightened. Quite often this action will throw off the alignment! To ensure that drive performance isn't adversely affected, the alignment has to be rechecked and even re-performed after the screws have been tightened. This additional step is absolutely crucial to attaining a high integrity alignment.

Unfortunately, this extra step can be very frustrating and time-consuming even for an experienced technician and is often omitted. As a result many technicians only lightly secure the screws — hence our term "light-screwing." Nearly two-thirds of the drives coming to us for the first time have improperly tightened head screws. If your drive fails every few months chances are that your service agency is "light-screwing" you and your heads. A lot of bad press has been generated against the PerSci drive as a result of this kind of service. However, I must emphasize that this is not the fault of the disk drive!

Q2. Can PerSci drives be repaired if the only problem is intermittent "non-booting?"

A2. Absolutely! However, I have often found that this problem is related to factors outside of the drive.

First check your boot disks. If you are like most of my clients, you may have many copies and some of these duplicates may be damaged. Disks don't last forever. I have best results using Dysan floppies and less luck with Scotch, 3M and Verbatim.

Another very common problem outside of the drive is bad D.C. power supply connections. The ten-pin Molex connector that attaches between the PerSci and Cromemco system uses nine single-leaf spring pins which connect with the male pins on the upper edge of the main logic board on the PerSci (pin 4 is the nylon key). Two extremely common conditions may exist that may interfere with making good electrical contact. They are corroded contacts and flattened pin springs. The symptoms seen by the user vary widely from no drive response accompanied by funny noises to occasional no read or boot. The quick and nearly permanent fix requires the removal of each pin from the Molex connector using a small blade screwdriver. Clean the flat surface with an ordinary pencil eraser and then manually spread the leaf spring to approximately ¼ of an inch at its end with the screwdriver. This process will require less than 10 minutes once you have gained access to the connector.

Another possible source of intermittent problems deals with the original writing of the disk. If the disk was written on a marginally aligned drive or even the same drive prior to its last alignment you may have this trouble.

The PerSci disk drive will read a disk up to +/- 30% alignment error (A.E.),

which is much better than the Tandon, Shugart and Qume drives that I have worked with. Ideally, your heads will be aligned to 100% or 0% alignment error. However, if, for example, your drive wrote to your disk at -20% A.E. before alignment and was then aligned to +15% A.E. (+/- 15 % A.E. is the minimum factory specification) you have a net 35% A.E. from when the disk was first written and problems may result. All drives serviced here are aligned to less than 5% A.E.

Fine tuning our alignments is much more time consuming but it pays off in greater customer satisfaction and far fewer warranty returns. If your drive is aligned this closely you will most likely be able to read any disk that was ever written on that or any drive so long as the disk remains intact.

Q3. How do you clean the heads?

A3. The best method is to manually clean the heads using a Q-tip and a solution of alcohol greater than 93%. To do this requires removing the drive from the system and actually observing the read/write surface as you clean. This is important as many operating environments are sufficiently unclean as to allow a build up of airborne contaminants on both the disk heads and the disk surfaces themselves. This build-up will ultimately cause bad reads and writes and may even result in scratching the surface of floppy disks used in that drive slot! One word of caution. Be extremely careful when using the Q-tips near the floating heads. If you slip or snag the corners of the head with the Q-tip you can crash the heads! So I do not recommend this procedure for casual or inexperienced technicians or users.

An alternative is to use the standard 8″ head cleaning kits that use a textured but non-abrasive "disk" that you physically moisten with alcohol just prior to insertion into the drive. These kits are available through most office supply catalogs for under $30, and are well worth it. I recommend using the kit one half minute per disk slot per month. Remember that the heads must be selected while cleaning so that both heads will be in contact with the cleaning disk. We use RDOS for "B", "C" and "D" heads and "B" for boot on "A" heads. Our in-house procedures include using the kit the first Monday morning of each month. In that way we are conditioned to keep our drives clean on a regular basis and this minimizes problems with our in-house drives. In fact, I have one PerSci drive that has worked without a single failure since 1981!

Q4. Will better booting result from using an updated version of RDOS?

A4. Not in my experience. However, versions above 2.53 will allow booting from any floppy drive connected to the system and provide enhanced diagnostics from RDOS.

Q5. Since PerSci is out of business, how will a service technician be able to get spare parts in the future?

A5. PerSci is NOT out of business! In fact they still manufacture and sell disk drives and spare parts! They were purchased several years ago by a company called CES of Hawthorne, California. Last time I dealt with them they had reorganized under the name of E.F. Industries. While the manufacture and sale of new equipment has been cut back dramatically, they continue to fill orders from manufacturers internationally. Per drive prices in excess of $2500 and lead times of up to six months do not make the machine competitive in the small systems arena, but several mini and mainframe manufacturers continue to install the PerSci into their respective environments. Spare parts availability and pricing are not what they used to be. We have found second sources for 90% of the parts needed to support the PerSci and maintain ample stocks of the remaining 10%.

Many Cromemco customers have been adversely influenced about the quality and reliability of the PerSci disk drive because of a few "horror" stories. Some of these contain a measure of truth but the overwhelming majority of these rumors are exaggerated. As mentioned earlier in this article, most of the first 1500 PerSci drives delivered into the Cromemco environment were "dogs!" This was Cromemco's first exposure to the PerSci 299B and although all the basic problems have been rectified, the fall-out has had a lasting effect on the drive's reputation.

Here at Peripheral Labs, we have been intimately involved with the support of the PerSci disk drive for many years and currently oversee the service of over 5000 units installed internationally.

Most of these machines require only infrequent service about every 12 to 18 months. The exceptions are industrial or similar environments where the input A.C. power is subject to wide fluctuations and where larger amounts of airborne contaminants are present. One particularly problematic situation involves the installation of the PerSci disk drive in mobile applications — we specifically discourage this practice.

Some of our more conscientious clients make a practice of sending their drives to us yearly for preventive maintenance and consequently enjoy greatly enhanced performance.

With good service, a clean operating environment and reasonable care your PerSci disk drive will provide you with many years of reliable high-speed performance.

About the Author:

John A. Bush *is Founder and President of Peripheral Labs (AKA P.P.S.) a specialized Southern California computer service firm. He is the former Supervisor and Lead Technician at PerSci, Inc., and has been working in the Cromemco system environment for over six years.*

Prior to his entering the microcomputer arena, he served 6 years with the U.S. Navy as an electronics technician specializing in the technical support of shipboard cryptographic communication systems. He holds degrees from Long Beach City College and Mount San Antonio College in California. He can be reached by telephone at (714) 861-6649.

Satellites

Continued from front cover

The article concludes with the plans for continued development of the systems to perform new functions such as a **gateway** to the JAS-1 store-and-forward packet radio satellite. December of 1986 marked the twenty-fifth anniversary of the launch of the first amateur radio satellite.

Introduction

It is fairly well known that one specialty area within the Amateur Radio Service is concerned with the design, construction, operation, and use of amateur radio satellites. Of course, the majority of amateur radio operators participate in this specialty as users of the communications **transponders** aboard the spacecraft. Design, construction, and operation is usually done through international cooperation of university groups and amateur radio operators who are employed in the aerospace industry. December of 1986 marked the twenty-fifth anniversary of the launch of the first amateur radio satellite, OSCAR-1. The designation OSCAR is usually carried as a part of the satellite name after launch and is an acronym for Orbiting Satellite Carrying Amateur Radio.

At first these satellites were very simple and their operational lifetimes were short. Later, the satellites became more sophisticated and carried command and control systems, **telemetry systems**, and communications transponders. However, the primary mission of amateur radio spacecraft remained one of providing a communications utility. This trend was due a change and along with the change came the establishment of onboard microprocessor systems.

In 1981, UoSAT-OSCAR-9, designed and constructed by the University of Surrey (England) Electrical Engineering Department, was launched. UoSAT-OSCAR-9 was the first amateur radio spacecraft to be dedicated to applied research in cost-effective spacecraft engineering and scientific experiments. It was also the first amateur radio spacecraft to carry onboard microprocessors as part of its command, control, and telemetry systems. Data is telemetered to the ground in ASCII at 1200 baud and is thus easily processable by a microcomputer system with the proper software. UoSAT-OSCAR-9 carries no communications transponder. The University of Surrey's second satellite, UoSAT-OSCAR-11, was launched in March, 1984. See Figure 1 for a system diagram of the UoSAT-OSCAR-11 spacecraft. UoSAT-OSCAR-9 and UoSAT-OSCAR-11 have come to be called UoSAT-1 and UoSAT-2 and will be referred to as such in this article.

After some additional background information, this article will describe two microcomputer systems designed to automatically track and collect data from amateur radio spacecraft, particularly the UoSATs. The additional background is needed to provide insight into the motivation for the development of the systems. The article will conclude with an evaluation of the systems and the plans for changing and expanding the functions of the systems. Since this article is intended to present the hardware-related aspects of the systems, software will not be discussed in great detail.

BACKGROUND

Spacecraft in Operation

UoSAT-1 and UoSAT-2 — These spacecraft have already been mentioned. They are both in polar, **sun-synchronous** orbits. UoSAT-1 has an **apogee** of approximately 500 km. and is usually accessible during two orbits occurring in the early morning hours (3:00 to 6:00 A.M.) and two orbits occurring in the middle afternoon hours (3:00 to 6:00 P.M.). UoSAT-2 has an apogee of approximately 700 km. and has roughly the same number of accessible orbits per day except that they occur during the late evening (8:00 to 11:00 P.M.) and late morning (9:00 A.M. to 12:00 noon) hours. The access time will vary up to a maximum of about 18 minutes. Local times are given in Central Standard Time. The obvious inconvenience of some of these times is one motivation for the development of an automated system.

The operating system software in these spacecraft is now quite sophisticated and various types of data will be captured even within a single access period. A typical combination might be plain text bulletins, telemetered data from spacecraft systems and experiments, whole-orbit telemetry surveys, and for UoSAT-2, the message headers resulting from messages stored in the Data Communications Experiment, DCE.

AMSAT-OSCAR-10 — This spacecraft is the result of an international effort primarily by radio amateurs in Germany and the United States. It is also microprocessor controlled but its mission is that of providing communications transponders on two different frequency bands. It is different from previous spacecraft in that its orbit is highly elliptical with an apogee of **geosynchronous** altitude (37,500 km.). In this orbit it provides as much as 12 hours per day access time to parts of the world. However, it should be noted this orbit is neither geosynchronous nor geostationary. It is of interest to this author not for its analog communications transponders but for its differing telemetry systems and data transmission format as well as the digital communications experiments that have been tried using its analog transponders.

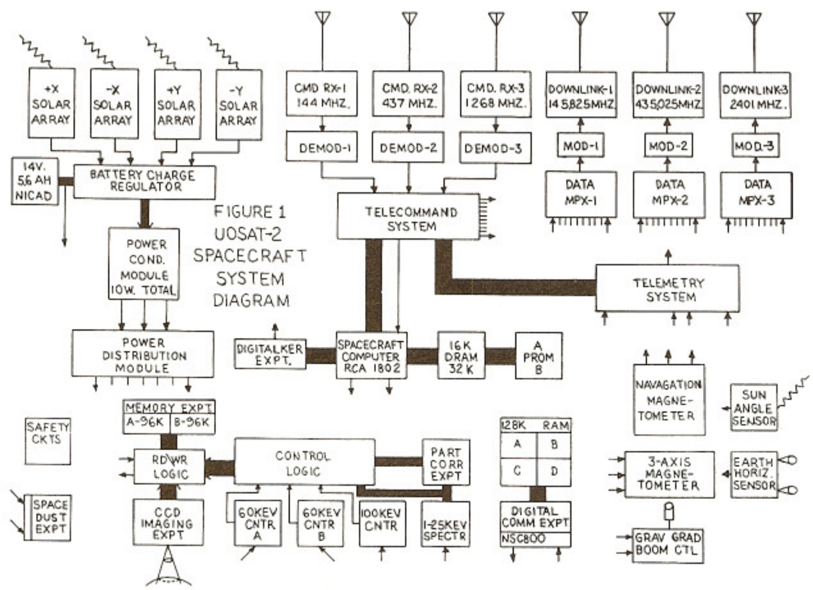


FIGURE 1 UoSAT-2 SPACECRAFT SYSTEM DIAGRAM

JAS-1 — The first amateur radio satellite with a free access store-and-forward digital communications transponder is called JAS-1. It is a product of Japanese origin where the transponders were built by Japanese radio amateurs and the spaceframe was built by NEC. It was launched in August, 1986 by NASDA (the Japanese equivalent of NASA). A properly equipped automated ground station could serve as a gateway to/from terrestrial packet radio networks and JAS-1.

Orbit Prediction

The prediction of the satellite access times for a given point on the earth is

a computation-intensive task. Yet, an automated system must have this data for every satellite it is to track. The algorithms used to generate this data for the systems described here were developed by Dr. Thomas A. Clark and were first published in 1980 [1]. His program may have been the first microcomputer-based prediction program using the standard Keplerian elements as the reference point for computations. Later this author implemented the algorithms on several different microcomputers for the benefit of amateur satellite enthusiasts [2,4]. More recently, the original algorithms have been enhanced by Dr. Robert W. McGwier to provide a closed-form rather than iterative solution to the problem of finding **AOS** and **LOS** time [8].

Hardware Selection

The final topic before presenting the two hardware configurations is that of hardware selection. The reader may wonder why the systems have been implemented on S-100 bus Z-80 computers. There are several reasons. First, amateur radio operators are always at the forefront of experimentation. For this reason many started their experimentation with microprocessors at a time when the S-100 bus was much more popular. Second, in any amateur radio project there is the very practical tendency to use the equipment one has available to minimize costs. Finally, it is now easy and inexpensive to acquire 8-bit microcomputers which are adequate for dedicated systems such as described here due to the trend to upgrade to newer 16- and 32-bit systems.

Single-Computer Control System

The first automated system developed is shown in Figures 2 and 3. Figure 2 concentrates on equipment external to the computer while Figure 3 provides additional detail about the internal computer system components. As can be seen, the system provides for control of antenna position during satellite access times, control of antenna circularity (polarization), control of radio receiver frequency to compensate for Doppler effect, and control of the tape recorder for both data capture and recording of the satellite/orbit/date/time stamp from the speech synthesizer. All of the above items were conceived in the initial system design with the exception of the speech synthesizer for voice identification of the recorded tapes. This was added as an afterthought. The pertinent details of the method of implementation of these features follow.

Antenna Positioning

The antenna position is updated during a satellite pass at a time interval corresponding to the one used when the orbital predictions were made. Several system components are involved.

First, a new set of azimuth and elevation headings are acquired at the start of a new time step. Next, from parameters known by the program, these headings are converted to the corresponding position indicating voltages to be read from the rotator control boxes by the analog-to-digital converter channels. Third, the current position voltages are read. By comparing the current and target voltages the required direction of movement is found. Fourth, movement in the proper direction is initiated by setting the proper bit of the latched parallel output port going to the rotator control interface. Finally, the motion is stopped when the position voltage read from the A-to-D, which is being read continuously, matches the target voltage.

Depending on the A-to-D used, rescaling of the input may be required so that the input range of the A-to-D is consistent with that produced by the rotator position indicator. Also, in some cases additional voltage regulation may be required for the power supply providing the position indication voltage. With a little work a combination can be found that allows the antennas to be positioned to within one unit on the analog position indicator. This is usually 5 degrees for typical azimuth rotators and 3 degrees for elevation rotators. This level of accuracy is adequate for most antennas found at amateur radio stations.

Antenna Polarization Switching

The signal as originated from the satellite is either right- or left-circular polarized. However, due to spacecraft orientation, spacecraft motion, propagation effects, and other factors, the signal-to-noise ratio may sometimes be improved by switching to the opposite polarization sense. This is especially true when the fade is deep and prolonged and not caused by local man-made noise.

To attempt to accommodate these fades, the circularity switch mounted at the antenna is wired to one of the relays on the 4PIO isolated interface. When there is no other function to perform the system constantly monitors the signal strength as indicated by the receiver S-meter. As long as the level stays above some threshold set in software nothing is changed. If the level does fall below the cutoff point nothing is done immediately but the system continues monitoring to see if the trend is either consistently downward or prolonged according to other software counters. If it is a consistent downward trend a switch is made to the opposite polarization.

Frequency Correction for Doppler Shift

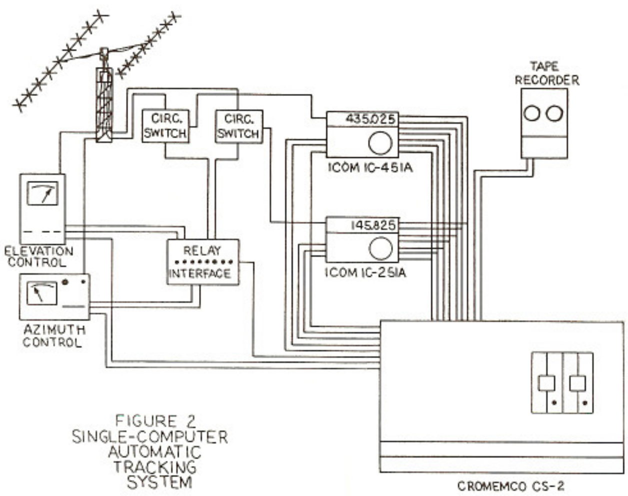


FIGURE 2
SINGLE-COMPUTER AUTOMATIC TRACKING SYSTEM

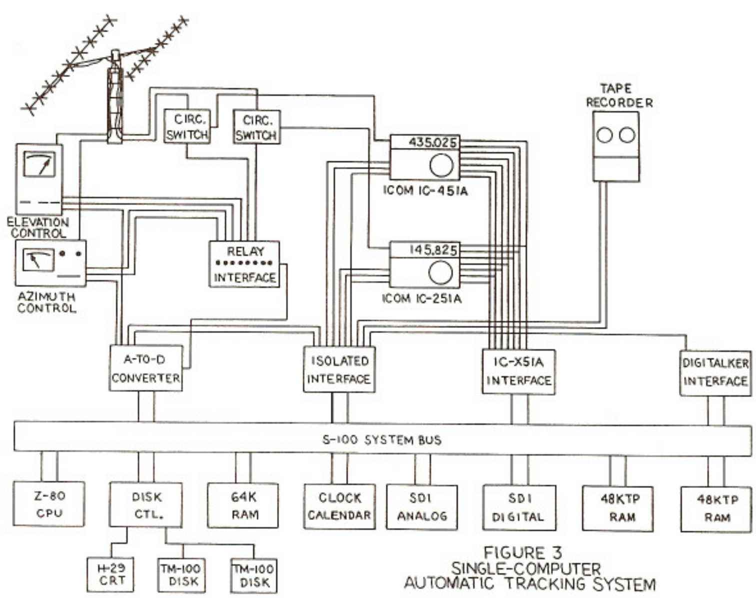


FIGURE 3
SINGLE-COMPUTER AUTOMATIC TRACKING SYSTEM

As is the case with any transmitting source in motion with respect to the observer, there is Doppler shift observed when receiving the signal from the satellite. Although it is not considerable at the usual receiving frequency of 145.825 MHz., the ability to automatically compensate for it was thought to be a desirable feature.

The implementation of this feature requires a custom-built interface between the S-100 bus and the microprocessor in the ICOM IC-251A receiver. The requirement is to decode a Z-80 port address and provide the data to the microprocessor in the radio at CMOS levels over a 4-bit data bus. There are also control signals for data transfer and other status indications such as signal/no signal being received. Thus, data must also be read from the receiver through a Z-80 port. The interface in use was designed by Curt Terwilliger of Cromemco, Inc. [10].

During a satellite pass, the receiver frequency is updated at each time step after antenna positioning. The frequency is first set using the Doppler computed during the orbit prediction phase. This is used as an initial guess after which the frequency is moved in the required direction in 100 Hz. steps while monitoring the center frequency (discriminator) meter output. Before the zeroing process is started, the metering circuit must be switched from signal strength to center frequency. It is returned to signal strength monitoring after the frequency is set.

Tape Logging and Control

The audio tape recorder used to capture the data is turned on and off by a relay on the 4PIO interface. It was soon learned that it would be much better if there was some sort of marker recorded on the tape between each data set.

The solution to this problem was another custom-built board interfacing the National Semiconductor Digitalker speech synthesizer chip set to the system. The circuit was designed by Dr. Aaron Brown and is documented in [5].

The control scheme was modified slightly such that at the beginning of a satellite pass the audio input of the tape recorder is switched to the Digitalker interface. Then, a voice label is placed on the tape consisting of the satellite identifier, orbit number, date and time under program control. After the label has been generated, the input to tape recorder is switched back to the receiver audio.

Software Issues

All of the previously described functions are controlled by a single computer program written in BASIC. The version of BASIC used was Cromemco 32K Structured BASIC. This dialect of BASIC is particularly powerful in that it provides all of the necessary structured programming features including procedures. This allowed easy development of a fairly complicated control system. There were no time-critical functions beyond the capabilities of an interpretive language.

Results of Actual System Operation

The single-computer control system has been in operation since October of 1983. As might be expected, various deficiencies were found and other desirable features were conceived. Some of these items would require software solutions while others would require a different hardware configuration.

The most serious deficiency of the system was choosing to tape record the data rather than capturing it directly on disk. Soon, large stacks of tapes accumulated which still had to be demodulated before the data could be inspected and processed. With two spacecraft operational providing, on the average, eight visible passes per day it is easy to see that one could spend days processing an accumulation of tapes. Clearly, this had to be changed.

Another problem was the inability to prioritize the tracking function among satellites. The desirability of this feature is best illustrated by an example. AMSAT-OSCAR-10 in its elliptical orbit may have a visibility time as long as 12 hours. In the mean time, one or more shorter duration passes of polar, sun-synchronous satellites such as UoSAT-1 and UoSAT-2 may have occurred and their data will have been lost. Even though OSCAR-10 does not carry scientific experiments, it is desirable to monitor its telemetry from time to time to see the status of the various spacecraft systems.

A minor fault of the system was noticed during the Space Shuttle missions which included amateur radio operations. The first of these was by Dr. Owen Garriot from Columbia in 1983. The problem arose due to the azimuth and elevation positioning being done serially rather than in parallel. The low altitude of the Space Shuttle orbit causes it to be moving very fast with respect to the ground station. On high-elevation passes the system had a hard time keeping up with the Space Shuttle.

Finally, some of the functions implemented could have been omitted without serious consequences. One of these was the Doppler shift correction function. Although it is still thought desirable it could be limited to using only the computed value to set the receive frequency. The zeroing to the on-the-air signal could be eliminated. This is particularly true on the two meter downlink frequency which is used the majority of the time. Another justification for the limitation of this feature is a present trend in modem design to allow the modem to control the receiver frequency directly. This is possible because almost all digital frequency control receivers have facilities for controlling the VFO from an external source.

Another feature that did not work well was the antenna polarization switching. Using the software approach described earlier, it was very difficult to decide when a switch would be beneficial. Frequently the switch would be made just when it would have been better to leave things as they were. The solution to this problem used by the University of Surrey ground station is to have two identical receiving systems with one connected to an antenna of RHCP and the other connected to an antenna of LHCP. The error rate is then monitored on both receive channels simultaneously and the system producing the lowest error rate is selected for data capture. For the systems described here, this approach would have been cost-prohibitive due to the duplication of equipment required.

This section has described the shortcomings of the single-computer system as observed through two years of actual operation. The experience gained provided the basis for development of the dual-computer system described next.

DUAL-COMPUTER CONTROL SYSTEM

The author is employed at Corpus Christi State University and due to the phase-out of Z-80 single-user systems, several donations of such equipment were made to the university. This turn of events coupled with the desire to remove deficiencies of the original system and also have a station in operation at the university caused the construction of a new system to begin in late 1985. The configuration to be described next was placed online on February 1, 1986 and has been in continuous operation since that time.

Two computers are employed in the new configuration. One of them is used for station control functions such as antenna positioning, radio control, and so forth while the other is used for data

capture. All data is held in memory until the satellite is no longer visible after which it is saved on disk. The station control computer sends commands to the data capture computer. Figure 4 shows equipment external to the two computers while Figure 5 provides additional detail about the internal computer system components.

Station Control System

The major changes in the station control system are software changes and will be discussed later. The antenna positioning function is handled in the same manner as the single-computer system. The system clock/calendar is an external rather than an internal device. Frequency control of the radio receiver, a Yaseu FRG-9600, is accomplished through a serial port provided on the receiver as standard equipment. However, this port operates at TTL rather than RS-232 levels. There is no attempt to zero the radio frequency to the actual received signal and there is no switching of antenna polarization. Finally, the two systems are connected via an 8-bit bi-directional parallel port for communication purposes.

Data Capture System

The major changes to the overall system come as a result of the addition of the second computer. As mentioned earlier, all of the captured data is held in memory until the spacecraft is no longer visible and then it is stored on disk. Since the Z-80 can only accommodate a 16-bit address bus and it is likely that more data than this will be captured on a single pass, bank switched memory is used.

Bank-Switched Memory — The data capture system has 3 banks of 32K bytes each for data storage. The bank switching is done by assigning one of the Z-80 I/O ports (40h) for this purpose. When a bank of memory fills, the program can issue an OUT instruction to bank-select port with the desired new bank number specified in the accumulator. The memory boards then decode this I/O instruction and those with a bank number matching the one specified are connected to the bus while all others are disconnected. If a memory board should remain connected to the bus at all times, it is wired to appear in all eight possible banks.

In the data capture system, addresses 0000 to 3FFFh are not switched since the data capture program resides there. Likewise, addresses C000 to FFFFh are not switched because the operating system resides in high memory. It is the 32K bytes occupying addresses 4000 to BFFFh that is bank switched during data capture operations.

While bank switching could be eliminated entirely by using a newer microprocessor with a larger linear address space, one must consider the cost of the computer system — in this case zero. The bank-switched memory carried a similar price tag.

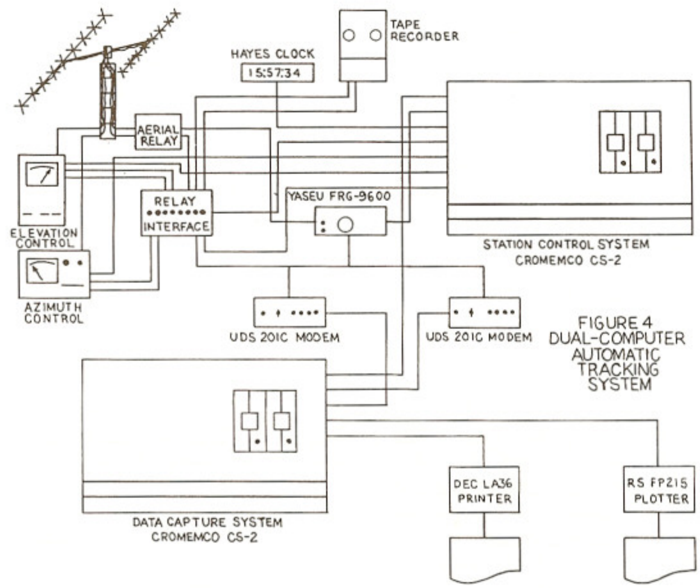


FIGURE 4
DUAL-COMPUTER
AUTOMATIC
TRACKING
SYSTEM

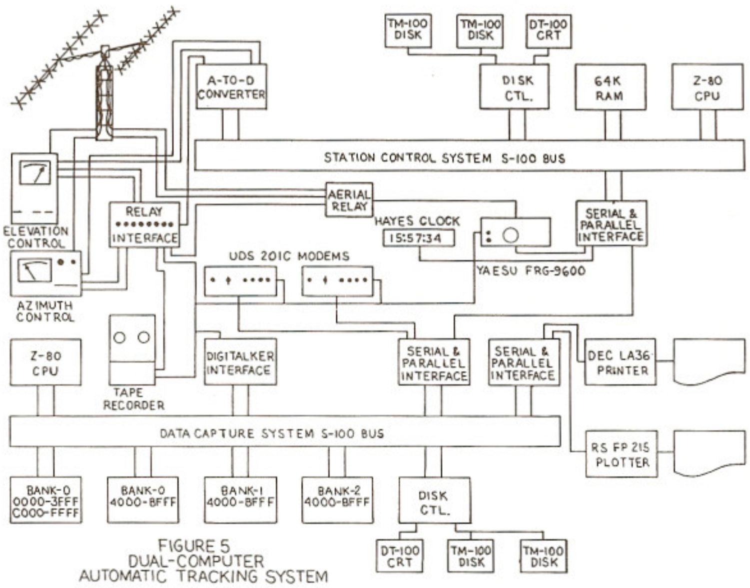


FIGURE 5
DUAL-COMPUTER
AUTOMATIC TRACKING SYSTEM

Multiple Modems — Since the data is now being demodulated in real time, there is a requirement to switch modems according to the modulation scheme used in the received data. This turns out to be one of the major assets of the new system both now and in the future. All satellites now being tracked use different modulation schemes. Furthermore, those already built and awaiting launch also use varying schemes. In some cases, the modulation scheme used on the uplink is different than the modulation used on the downlink.

Any required modems can be connected to the data capture computer subject only to the availability of serial interface cards. The proper modem is selected at data capture time by command sent from the station control computer according to specifications made by the operator for the particular satellite.

Software Support — Software running in the data capture computer is written in Z-80 assembly language. The program is one that evolved from one used previously for the separate demodulation and storage of the recorded data tapes. The program was modified so that it can be operated in both local and remote command mode. Remote command being the mode in which it will accept commands from the

station control computer. The station control computer can issue commands to select the desired receive port, and thus the correct modem, to start and stop data capture, and write the captured data to disk with the proper file name.

There are also several other new software features. One of them allows threshold values to be set for starting and suspending data capture. The start threshold requires that a certain number of consecutive error-free characters be received before data capture is started. The suspend threshold causes a certain number of consecutive character errors to stop data capture. Capture is resumed again when the start threshold criteria is fulfilled. This feature has the effect of eliminating the capture of many erroneous characters, particularly when signal strength is poor such as when the satellite is near the horizon at the beginning and end of a pass. Finally, the operator may specify the memory bank numbers and addresses to be used in the banked-memory capture process.

EXTENSIONS TO CURRENT SYSTEMS

Both of the systems described in this article are operational. The dual-computer system has proven entirely satisfactory from a hardware viewpoint. The single-computer system is used mostly for backup and experimentation. This is primarily due to the university location of the dual-computer system being a much more RF-noise-free environment. Also, the operation of the university system as the primary system serves to expose students to computer applications in the space sciences. This section briefly details some changes and extensions planned for both systems.

Single-Computer System

The hardware will be expanded to support the two antenna systems already in place. This will allow the tracking function for some satellites to be assigned to one antenna system and other satellites to a different antenna system. The control system software will be converted from BASIC to a compiled language, probably PL/I. In the diagram shown in Figure 3, the SDI graphics interface can be seen. This is not currently used but the plan is to utilize the graphics system to provide a real-time ground track display for the satellites being tracked.

Dual-Computer System

This system may also be expanded to support control of dual antenna systems but since the antenna systems are not already in place this has not been decided. However, the control program will be converted to a compiled language as well. At the same time a few minor problems will be fixed. The most significant change to this system will involve the integration of additional hardware and software so that it can serve as a gateway to the JAS-1 satellite.

The JAS-1 satellite was mentioned in the background section and was successfully launched during preparation of this article. For a description of the JAS-1 satellite systems see [11]. Complete details of the issues involved in integrating the gateway function are beyond the scope of this article. Briefly, though, the work will involve the integration of a typical terrestrial packet radio mailbox system such as shown in Figure 6. The details of the terminal node controllers are shown in Figure 7.

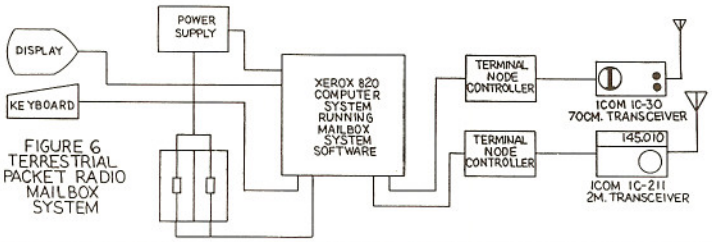


FIGURE 6
TERRESTRIAL PACKET RADIO MAILBOX SYSTEM

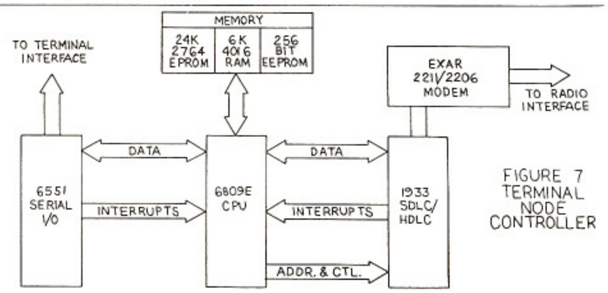


FIGURE 7
TERMINAL NODE CONTROLLER

Messages from such a system that are marked for uploading to the spacecraft would have to be fetched by the data capture computer and transmitted during a JAS-1 pass. The reverse process would also have to be done — retrieving data from JAS-1 destined for the local area network. For a fairly recent general description of packet radio operations in the Amateur Radio Service see [7].

CONCLUSION

The design, development, and implementation of the systems described here has shown that it is indeed possible to construct a sophisticated microcomputer-based system for low-earth-orbit satellite tracking and data collection. Even though the systems have been designed with the requirements of satellites operated in the Amateur Satellite Service in mind, they could be adapted for use with other spacecraft.

It remains to be seen if system functions can be expanded to include that of a packet radio satellite gateway for use with the JAS-1 satellite. The outcome of that endeavor will be reported in a future article.

ACKNOWLEDGEMENTS

The author would like to thank those individuals who contributed in various ways to the development of the systems described in this article. Mr. Carl Crownover, President of Jordan Laboratories, donated some of the hardware used in the station control computer of the dual-computer system. Dr. F. Michael Speed, President of Speedman Oil Company, donated the data capture computer system. Finally, thanks to Dr. Ruth Bakke, Dean of the College of Science and Technology, Corpus Christi State University, for providing the additional laboratory space for that part of the development work done at the university.

About the Author:

Robert J. Diersing *is an Associate Professor in Computer Science at Corpus Christi State University in Texas. An avid user of Cromemco equipment since their inception, he still maintains a Cromemco Z-1 computer running Dazzler graphics. Questions or comments regarding this article can be directed to him at the following address:*

Robert J. Diersing
Computer Science Department
Corpus Christi State University
6300 Ocean Drive
Corpus Christi, TX 78412
(512) 991-6810 ext. 476

---

capture. All data is held in memory until the satellite is no longer visible after which it is saved on disk. The station control computer sends commands to the data capture computer. Figure 4 shows equipment external to the two computers while Figure 5 provides additional detail about the internal computer system components.

Station Control System

The major changes in the station control system are software changes and will be discussed later. The antenna positioning function is handled in the same manner as the single-computer system. The system clock/calendar is an external rather than an internal device. Frequency control of the radio receiver, a Yaseu FRG-9600, is accomplished through a serial port provided on the receiver as standard equipment. However, this port operates at TTL rather than RS-232 levels. There is no attempt to zero the radio frequency to the actual received signal and there is no switching of antenna polarization. Finally, the two systems are connected via an 8-bit bi-directional parallel port for communication purposes.

Data Capture System

The major changes to the overall system come as a result of the addition of the second computer. As mentioned earlier, all of the captured data is held in memory until the spacecraft is no longer visible and then it is stored on disk. Since the Z-80 can only accommodate a 16-bit address bus and it is likely that more data than this will be captured on a single pass, bank switched memory is used.

Bank-Switched Memory — The data capture system has 3 banks of 32K bytes each for data storage. The bank switching is done by assigning one of the Z-80 I/O ports (40h) for this purpose. When a bank of memory fills, the program can issue an OUT instruction to bank-select port with the desired new bank number specified in the accumulator. The memory boards then decode this I/O instruction and those with a bank number matching the one specified are connected to the bus while all others are disconnected. If a memory board should remain connected to the bus at all times, it is wired to appear in all eight possible banks.

In the data capture system, addresses 0000 to 3FFFh are not switched since the data capture program resides there. Likewise, addresses C000 to FFFFh are not switched because the operating system resides in high memory. It is the 32K bytes occupying addresses 4000 to BFFFh that is bank switched during data capture operations.

While bank switching could be eliminated entirely by using a newer microprocessor with a larger linear address space, one must consider the cost of the computer system — in this case zero. The bank-switched memory carried a similar price tag.

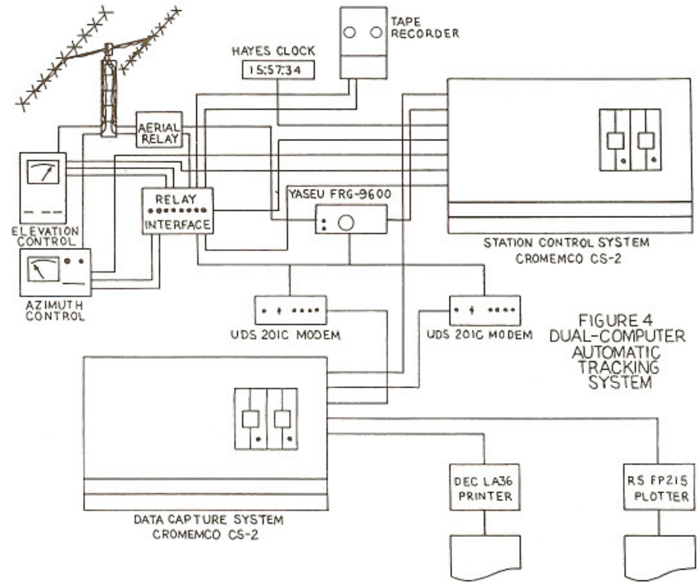


FIGURE 4
DUAL-COMPUTER
AUTOMATIC
TRACKING
SYSTEM

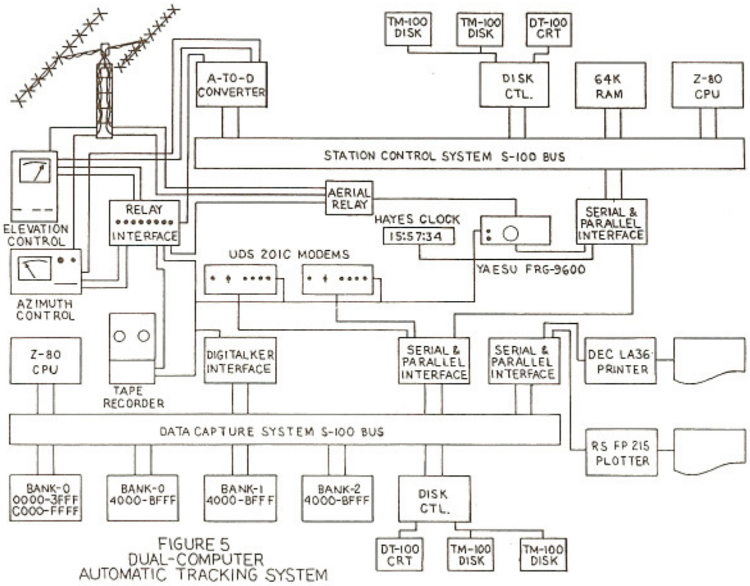


FIGURE 5
DUAL-COMPUTER
AUTOMATIC TRACKING SYSTEM

Multiple Modems — Since the data is now being demodulated in real time, there is a requirement to switch modems according to the modulation scheme used in the received data. This turns out to be one of the major assets of the new system both now and in the future. All satellites now being tracked use different modulation schemes. Furthermore, those already built and awaiting launch also use varying schemes. In some cases, the modulation scheme used on the uplink is different than the modulation used on the downlink.

Any required modems can be connected to the data capture computer subject only to the availability of serial interface cards. The proper modem is selected at data capture time by command sent from the station control computer according to specifications made by the operator for the particular satellite.

Software Support — Software running in the data capture computer is written in Z-80 assembly language. The program is one that evolved from one used previously for the separate demodulation and storage of the recorded data tapes. The program was modified so that it can be operated in both local and remote command mode. Remote command being the mode in which it will accept commands from the

station control computer. The station control computer can issue commands to select the desired receive port, and thus the correct modem, to start and stop data capture, and write the captured data to disk with the proper file name.

There are also several other new software features. One of them allows threshold values to be set for starting and suspending data capture. The start threshold requires that a certain number of consecutive error-free characters be received before data capture is started. The suspend threshold causes a certain number of consecutive character errors to stop data capture. Capture is resumed again when the start threshold criteria is fulfilled. This feature has the effect of eliminating the capture of many erroneous characters, particularly when signal strength is poor such as when the satellite is near the horizon at the beginning and end of a pass. Finally, the operator may specify the memory bank numbers and addresses to be used in the banked-memory capture process.

EXTENSIONS TO CURRENT SYSTEMS

Both of the systems described in this article are operational. The dual-computer system has proven entirely satisfactory from a hardware viewpoint. The single-computer system is used mostly for backup and experimentation. This is primarily due to the university location of the dual-computer system being a much more RF-noise-free environment. Also, the operation of the university system as the primary system serves to expose students to computer applications in the space sciences. This section briefly details some changes and extensions planned for both systems.

Single-Computer System

The hardware will be expanded to support the two antenna systems already in place. This will allow the tracking function for some satellites to be assigned to one antenna system and other satellites to a different antenna system. The control system software will be converted from BASIC to a compiled language, probably PL/I. In the diagram shown in Figure 3, the SDI graphics interface can be seen. This is not currently used but the plan is to utilize the graphics system to provide a real-time ground track display for the satellites being tracked.

Dual-Computer System

This system may also be expanded to support control of dual antenna systems but since the antenna systems are not already in place this has not been decided. However, the control program will be converted to a compiled language as well. At the same time a few minor problems will be fixed. The most significant change to this system will involve the integration of additional hardware and software so that it can serve as a gateway to the JAS-1 satellite.

The JAS-1 satellite was mentioned in the background section and was successfully launched during preparation of this article. For a description of the JAS-1 satellite systems see [11]. Complete details of the issues involved in integrating the gateway function are beyond the scope of this article. Briefly, though, the work will involve the integration of a typical terrestrial packet radio mailbox system such as shown in Figure 6. The details of the terminal node controllers are shown in Figure 7.

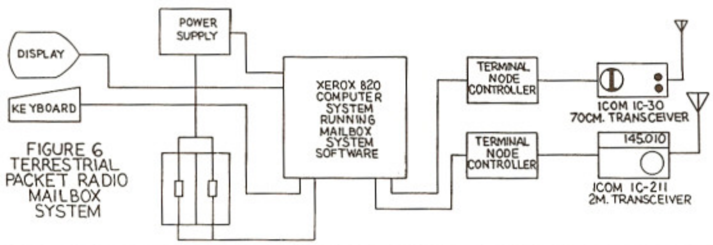


FIGURE 6 TERRESTRIAL PACKET RADIO MAILBOX SYSTEM

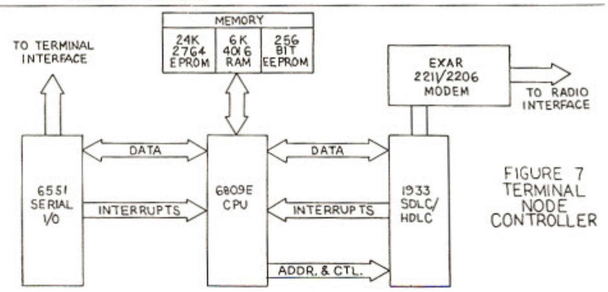


FIGURE 7 TERMINAL NODE CONTROLLER

Messages from such a system that are marked for uploading to the spacecraft would have to be fetched by the data capture computer and transmitted during a JAS-1 pass. The reverse process would also have to be done — retrieving data from JAS-1 destined for the local area network. For a fairly recent general description of packet radio operations in the Amateur Radio Service see [7].

CONCLUSION

The design, development, and implementation of the systems described here has shown that it is indeed possible to construct a sophisticated microcomputer-based system for low-earth-orbit satellite tracking and data collection. Even though the systems have been designed with the requirements of satellites operated in the Amateur Satellite Service in mind, they could be adapted for use with other spacecraft.

It remains to be seen if system functions can be expanded to include that of a packet radio satellite gateway for use with the JAS-1 satellite. The outcome of that endeavor will be reported in a future article.

ACKNOWLEDGEMENTS

The author would like to thank those individuals who contributed in various ways to the development of the systems described in this article. Mr. Carl Crownover, President of Jordan Laboratories, donated some of the hardware used in the station control computer of the dual-computer system. Dr. F. Michael Speed, President of Speedman Oil Company, donated the data capture computer system. Finally, thanks to Dr. Ruth Bakke, Dean of the College of Science and Technology, Corpus Christi State University, for providing the additional laboratory space for that part of the development work done at the university.

About the Author:

Robert J. Diersing *is an Associate Professor in Computer Science at Corpus Christi State University in Texas. An avid user of Cromemco equipment since their inception, he still maintains a Cromemco Z-1 computer running Dazzler graphics. Questions or comments regarding this article can be directed to him at the following address:*

Robert J. Diersing
Computer Science Department
Corpus Christi State University
6300 Ocean Drive
Corpus Christi, TX 78412
(512) 991-6810 ext. 476

REFERENCES

1. Clark, Thomas A. "BASIC Orbits," *ORBIT* Magazine No.6, AMSAT-USA, March, 1980, Page 6.

2. Diersing, Robert J., *Using Microcomputer Programs for Radio Amateur Satellite Orbital Prediction*, AMSAT-USA, October, 1983.

3. _______, "Microcomputer Processing of UoSAT-OSCAR-9 Telemetry," *QST*, August, 1984, page 23.

4. _______, "Applications of Microcomputers in the Amateur Radio Space Program," *South Central Regional ACM Conference*, Austin, Texas, November, 1984.

5. _______, "Notes on Microcomputer Applications in Automatic Station Control and Telemetry Data Processing for UoSAT-OSCAR-9 and UoSAT-OSCAR-11," *Second Annual Amateur Radio Satellite Symposium*, Los Angeles, California, November, 1984.

6. _______, *Microcomputer Applications in Station Control and Processing for UoSAT-OSCAR-9 and UoSAT-OSCAR-11*, AMSAT-UK, July 1985.

7. Karn, Phil R., Price, Harold E., and Diersing, Robert J., "Packet Radio in the Amateur Service," *IEEE Journal on Selected Areas in Communications*, May, 1985, Page 431.

8. McGwier, Robert W., "A Non-Cosmetic Improvement to the W3IWI Tracking Algorithm," Unpublished manuscript, July, 1986.

9. Miller, James, "AMSAT-UK JAS-1 Modem," *OSCAR NEWS* No. 60, AMSAT-UK, July, 1986, Page 31.

10. Terwilliger, Curt, "Computer Control of the IC-255A," *QST*, May, 1981, Page 30.

11. Yamashita, Fugio, "Outline of Satellite JAS-1," *Proceedings of the ARRL Amateur Radio 5th Computer Networking Conference*, Orlando, Florida, March 9, 1986, Page 122.

GLOSSARY OF TERMS

AOS — acronymn for Acquisition Of Signal.

apogee — The point in an orbit which is farthest from the earth; opposite of *perigee* (closest to earth).

gateway — the node or location in an earth-based network responsible for the collection of and transmission to satellite of data from other nodes in the network; data concentrator.

geo-synchronous — An orbit wherein the satellite maintains a fixed position with respect to a given location on earth, i.e., the satellite appears to be stationary with respect to that location.

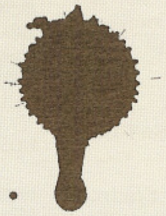
LOS — acronymn for Loss Of Signal.

sun-synchronous — Indicates that a given location on earth has access to a satellite at approximately the same time(s) each day, i.e., the satellite is moving with respect to that location.

telemetry — Use of electrical apparatuses which can indicate or measure various quantities and transmit the data to a distant point.

transponder — A device that receives a signal from one telecommunication circuit and transmits the corresponding signal to another circuit.

CURSES!!



(not something said at 2am.)

by Colin "Soup" Campbell

In my last article I spoke briefly about the termcap data base and the C routines commonly used to access that data base. At that time I promised to discuss the UNIX library of functions known as CURSES. CURSES provide the screen drivers for programs that are terminal intensive and terminal independent, such as screen editors. The author of CURSES is Ken Arnold who wrote the package while at Berkeley. What can the CURSES Library do?

- Move the cursor to any point on the screen.
- Insert or overlay text anywhere on the screen, even in reverse video (highlighted).
- Divide the screen into smaller rectangular areas.
- Manage each WINDOW independently. Scroll one while erasing another.
- Draw a box around a WINDOW using variable characters.

The basic idea behind CURSES is that you view through a WINDOW which is an array of characters representing the number of rows and columns possessed by your terminal. The size is fixed when the program executes, and is based upon the information found in the termcap data base. The basic WINDOW is called stdscr—standard screen—or terminal. This is referenced by other working WINDOWS. Each working window updates the stdscr only when a refresh command is issued. Working windows can be full windows and either overlay, or overwrite stdscr or they can be *subwindows* (small pieces of a bigger window). Subwindows are commonly used in packages that fork a shell process, a prime example being Microsoft Windows, or Side Kick.

Functions & Macros in my CURSES library:

```
waddch(win,c)          /* add character to current position */
waddstr(win,str)       /* add string starting at current position */
box(win, vert, hor)    /* draw a box around window using vert & hor characters */
wclear(win)            /* clear the window */
wclrtobot(win)         /* erase everything from current position to end of window */
wclrtoeol(win)         /* erase everything from current position to end of line */
mvcur(oldy,oldx,newy,newx)  /* move physical cursor */
gettmode()                /* get tty status */
setterm(type)             /* set term capabilitys from /etc/termcap */
wdelch(win)               /* delete single character & shift line left */
wdeleteln(win)            /* delete line & shift up  curx & cury unchanged */
delwin(win)               /* delete window & free memory to system */
endwin(win)               /* reset tty to entry modes */
werase(win)               /* erase everything on the window */
wgetch(win)               /* get a char from keyboard & echo it to window */
wgetstr(win,str)          /* calls wgetch(win) until \n is received */
initscr()                 /* initialize the current & standard screen */
winsch(win,c)             /* insert char & leave curx & cury unchanged */
winsertln(win)            /* insert char & leave curx & cury unchanged */
longname(bp, dev)         /* fill def with long name of the terminal */
wmove(win,y,x)            /* move the cursor to y,x */
mvprintw(y,x,fmt,args)    /* move to sprintf into stdscr */
mvwprintw(win,y,x,fmt,args)      /* move to sprintf into window */
mvscanw(y,x,fmt,args)     /* move and sscanf(stdscr, fmt, args)   */
mvwscanw(win,y,x,fmt,args)       /* move ans sscanf(win, fmt, args) */
mvwin(win,by,bx)          /* relocate starting position of window */
newwin(num_lines, num_cols, begy, begx)
                          /* allocate space & defaults for win */
subwin(orig, num_lines, num_cols, begy, begx)
                          /* make window within a window */
overlay(win1, win2)       /* write win1 on win2 non-destructively */
overwrite(win1,win2)      /* write win1 on win2 destructivly */
printw(fmt,args)          /* printf on stdscr */
wprintw(win,fmt,args)     /* sprintf on window */
wrefresh(win)             /* update current screen to look like win over */
                          /* the area covered by win.  Take note : this is */
                          /* very much optomized, and frequently it is good */
                          /* to use touchwin(win) before wrefresh(win)    */
scanw(fmt,args)           /* sscanf on standard screen */
wscanw(win,fmt,args)      /* sscanf on window */
scroll(win)               /* scroll the window up 1 line */
wstandout(win)            /* start entering characters as inverse video */
wstandend(win)            /* end entering characters as inverse video */
touchwin(win)             /* mark entire window to be updated at next refresh */

/*
 * psuedo functions for standard screen
 */
#define addch(ch)       VOID(waddch(stdscr,ch))
#define getch()         VOID(wgetch(stdscr))
#define addstr(str)     VOID(waddstr(stdscr, str))
#define getstr(str)     VOID(wgetstr(stdscr, str))
#define move(y,x)       VOID(wmove(stdscr,y,x))
#define clear()         VOID(wclear(stdscr))
#define erase()         VOID(werase(stdscr))
#define clrtobot()      VOID(wclrtobot(stdscr))
#define clrtoeol()      VOID(wclrtoeol(stdscr))
#define insertln()      VOID(winsertln(stdscr))
#define deleteln()      VOID(wdeleteln(stdscr))
#define refresh()       VOID(wrefresh(stdscr))
#define inch()          VOID(winch(stdscr))
#define insch(c)        VOID(winsch(stdscr,c))
#define delch()         VOID(wdelch(stdscr))
#define standout()      VOID(wstandout(stdscr))
#define standend()      VOID(wstandend(stdscr))

/*
 * mv function
 */
#define mvwaddch(win,y,x,ch)    VOID(wmove(win,y,x)==ERR?ERR:waddch(win,ch))
#define mvwgetch(win,y,x)       VOID(wmove(win,y,x)==ERR?ERR:wgetch(win))
#define mvwaddstr(win,y,x,str)  VOID(wmove(win,y,x)==ERR?ERR:waddstr(win,str))
#define mvwgetstr(win,y,x,str)  VOID(wmove(win,y,x)==ERR?ERR:wgetstr(win,str))
#define mvwinch(win,y,x,str)    VOID(wmove(win,y,x)==ERR?ERR:winch(win))
#define mvwdelch(win,y,x)       VOID(wmove(win,y,x)==ERR?ERR:wdelch(win))
#define mvwinsch(win,y,x,c)     VOID(wmove(win,y,x)==ERR?ERR:winsch(win,c))
#define mvaddch(y,x,ch)         mvwaddch(stdscr,y,x,ch)
#define mvgetch(y,x)            mvwgetch(stdscr,y,x)
#define mvaddstr(y,x,str)       mvwaddstr(stdscr,y,x,str)
#define mvgetstr(y,x,str)       mvwgetstr(stdscr,y,x,str)
#define mvinch(y,x)             mvwinch(stdscr,y,x)
#define mvdelch(y,x)            mvwdelch(stdscr,y,x)
#define mvinsch(y,x,c)          mvwinsch(stdscr,y,x,c)

/*
 * psuedo functions
 */
#define clearok(win,bf)         (win->_clear = bf)
#define leaveok(win,bf)         (win->_leave = bf)
#define scrollok(win,bf)        (win->_scroll = bf)
#define flushok(win,bf) (bf?(win->_flags|=_FLUSH):(win->_flags &=~_FLUSH))
#define getyx(win,y,x)  y=win->_cury,x=win->_curx
#define winch(win)      (win->_y[win->_cury][win->_curx] & 0177)

#include <modeequ.h>
#define XTABS XTAB
#define CRMOD CRDEVICE
#define raw() (setmode(_tty_ch,MD_MODE1,RAW,RAW),_rawmode=TRUE)
#define noraw() (setmode(_tty_ch,MD_MODE1,0,RAW),_rawmode=FALSE,nl())
#define crmode() (setmode(_tty_ch,MD_MODE3,CBREAK,CBREAK))
#define nocrmode() (setmode(_tty_ch,MD_MODE3,0,CBREAK),nl())
#define echo() (setmode(_tty_ch,MD_MODE1,ECHO,ECHO),_echoit=TRUE)
#define noecho() (setmode(_tty_ch,MD_MODE1,0,ECHO),_echoit=FALSE)
#define nl() (setmode(_tty_ch,MD_MODE1,CRDEVICE,CRDEVICE))
#define nonl() (setmode(_tty_ch,MD_MODE1,0,CRDEVICE))
#define bin() (setmode(_tty_ch,MD_MODE3,BINARY,BINARY))
#define nobin() (setmode(_tty_ch,MD_MODE3,0,BINARY),nl())
```

Further information may be obtained from:

Programming with Curses, InfoPro Systems, 3108 Route 10, Denville, NJ 07384, (201)989-0570, Price $8.

Screen Updating & Cursor Movement Optimization: A Library Package, Ken Arnold, Berkeley.

About the Author:

Colin "Soup" Campbell *is owner of Soup's Electronics in Fairbanks, Alaska. He has been involved with Cromemco equipment and programmng since 1981. Questions or comments regarding this article can be sent to:*

Colin Campbell
Soup's Electronics
P.O. Box 81871
College Branch
Fairbanks, AK 99708
(907) 456-3487

SOFT TIPS

SOFT TIPS is a regular column aimed at providing software oriented hints and ideas. Member contributions are encouraged. **SOFT TIPS** is edited by Robert Brown, President of **Excalibur Computers**, 4548 Auburn Blvd., Suite 191, Sacramento, CA 95841. Mr. Brown can be reached at (916) 971-9610.

Data Entry in Programs

Oftentimes, data entry causes many problems in programs. There are so many things to check for — was the right character entered, did they enter too many characters, was a character required, is the data on the correct place on the screen, what about the default values, and so on.

Below is a general purpose 'C' data entry routine for strings. The format for usage would be:

cc = input(row, col, string, buf, maxlen, defstring)

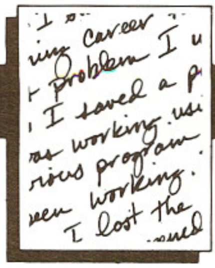
This would position the information, put up a prompt, guarantee no more than a certain number of characters and allow a default to be entered. In addition, it allows the use of backspace, delete and Control-U (to clear the line). Underscores are printed on the screen to let the user know how long the input is. It can be easily modified to include help information, and even specialized to work for SSNs, dates, money, and other special format type data.

```
/*  define for later use in the input function   */

#define EOS    '\0'
static char FLDCHR[] =
"______________________________________________________________________________";

int input(row, col, string, buf, maxlen, defstring)
int row, col, maxlen;                   /*  row, col = Cursor start position */
char *string, *buf, *defstring;         /*  string   = Prompting information
                                            buf      = result buffer
                                            maxlen   = maximum characters to
                                                       be entered
                                            defstring= default data         */
{
 int len, ch, i, j, dlen, uflag;        /*  various temporary variables     */

 buf[0] = EOS;                          /*  empty out the result buffer     */
 len = strlen (string);                 /*  find out how long the prompt
                                            string is.                      */

 dlen= strlen (defstring);              /*  find out how long the default
                                            output data is                  */
 pos (row, col);                        /*  places the cursor at row, col   */
 printf("%s%s", string,defstring);      /*  print the prompt, followed by
                                            the default data                */

 write(STDOUT, FLDCHR, maxlen-dlen);    /*  Put up empty underscores if the
                                            default data does not fill up
                                            the maximum number of chars     */

 pos (row, col + len);                  /*  Position the cursor at the start
                                            of the default data             */

 i = 0;                                 /*  initialize the character count  */
 uflag = FALSE;                         /*  uflag TRUE means a control-U    */
                                        /*  was pressed                     */
 while ((ch = grab ()) != '\n')         /*  keep getting characters until


                                         a newline is received.  Under
                                         CDOS, you must also test for
                                         a '\r'.                        */
{
 switch (ch)
  {
                                                /* if at the start of a  */
                                                /* new string, beep      */
   case 127:                                    /* if the character is a */
   case 8:        if (i == 0)   beep();         /* delete or a back space */
                  else                          /* back the cursor up,    */
                    {                           /* print the underscore   */
                     buf[i--] = NULL;           /* and back the cursor up */
                     wrbyte(STDOUT, 8);         /* again.                 */
                     wrbyte(STDOUT,'_');
                     wrbyte(STDOUT, 8);
                    }
                  buf[i] = EOS;                 /* set the end of the string */
                  break;                        /* leave and get next char */

   case '\025':   buf[0] = EOS;                 /* clear the result buffer */
                  pos (row, col + len);         /* to nothing, set the     */
                  write(STDOUT, FLDCHR, maxlen) ;  /* clear the screen to  */
                  pos (row, col + len);         /* underscores and place   */
                  i = 0;                        /* at data entry position  */
                  uflag = TRUE;                 /* Indicate that control-U */
                  break;                        /* has been pressed        */
                                                /* leave and get 1st char  */

   case 27:       return (-1);                  /*  if an escape is pressed */
                  break;                        /*  leave the routine       */
                                                /*  result buffer will have */
                                                /*  what was in it at the   */
                                                /*  time of the Escape      */

   case '?':      help();                       /*  run help routines       */
                  break;

    default:      if (ch >= ' ' && ch <= '~' && i < maxlen)
                    {                           /*  make sure the character */
                     wrbyte(STDOUT, ch);        /*  in a valid set and put  */
                     buf[i++] = ch;             /*  in on the screen and in */
                     buf[i] = EOS;              /*  the result buffer       */
                    }
                  else
                     beep();                    /*  Either an invalid char  */
                                                /*  or attempting to go too */
                                                /*  many characters         */

    }
  }
 if (buf[0] == EOS && uflag == FALSE)           /* if the result buffer    */
  {                                             /* is empty, assign it the */
   strcpy(buf, defstring);                      /* default string.  uflag  */
   pos(row, col + len + dlen);                  /* would indicate a null   */
   i = dlen;                                    /* desired and not defstring */
  }

 for (j = i; j < maxlen; ++j)                   /* clear all unused spaces */


    wrbyte(STDOUT, ' ');                        /* the output              */
  return(i+1);                                  /* return number of bytes  */
}


grab()
{
/* this is used in input, to grab a single character without echoing  */
/* You can use rcdos () for CDOS systems and look up appropriate info */
/* regarding what to use for non-Cromemco systesm                          */

 int c;

/* Set for return w/o Carriage return and echo off */
/* Set Abort enable off */
/* Set immediate echo off */

 setmode (STDIN, 2, 0x20, 0x20 | 0x8);
 setmode (STDIN, 4, 0, 0x10 | 0x02);

 c = rdbyte(STDIN);

/* Reset to return with carriage return and echo on */
/* Set Abort enable on */
/* Set immediate echo on */
 setmode (STDIN, 2, 0x8, 0x20 | 0x8);
 setmode (STDIN, 4, 0x10, 0x10 | 0x02);

 return(c);

}

strlen(s)
char *s;

{
 char *p = s;

 while (*p != EOS)
   p++;

 return(p-s);
}

help ()
{
 /* This routine would print out help information.  The help data would
    apply specifically to this data entry.  If implemented, it would be
    necessary to pass a parameter such a the number of the help file to
    be printed.  The information could be stored in either memory or on
    disk
 */
}

Note:  wrbyte (STDOUT, ch) can be replaced on Non-Cromix Systems with
       write (STDOUT, &buf, 1) where buf is defined as an array of
       chars and buf[0] = ch.
```

Bugs Report

Version 2.14 of **68000 C** does not appear to support all of the fopen() features. In particular, it seems unable to open a file for both read and write access at the same time. Many public domain programs that I receive use fopen() rather than open() because of the buffering capabilities. Since Cromemco hardware already does buffering, it is not really necessary to implement it in software. Instead, use the following to write to a file:

```
pfd = open (filename, UPDATE);
sprintf(xbuf, "What ever data = %d  %d  %d\n", a, b, c);
cc  = write (pfd, xbuf, strlen(xbuf));
close (pfd);
```



USER NOTES

USER NOTES are useful techniques, tips, ideas and other helpful information gleaned from our member's experiences with their Cromemco systems. If you have something along these lines that you want to share, write it up and send it to I/O NEWS, c/o USER NOTES, PO BOX 17658, IRVINE, CA 92713.

Editor's Note:

The following helpful notes were contributed by IACU member Carl Wick, who thought it might be nice if there were sets of simple to follow, step-by-step instructions for performing various tasks under Cromemco's port of UNIX System V. Carl writes:

CDOS-to-UNIX File Transfers

The following information regards transferring files from a CDOS formatted disk to Unix using Cromix as a stepping stone. **Note:** std? is to be replaced with the transfer area used by your disk. Examples: 50 meg disk uses std4; 140 meg disk uses std7.

Step 1: Boot Cromix Operating System
Step 2: Move to directory to place file(s), e.g., /tmp:
d /tmp
Step 3: Copy file(s) from CDOS disk in drive A to current directory:
cdoscopy /dev/sfda "file name" or "*.*"
Step 4: Remove any NULL characters at the end of the file(s) using screen or ce under Cromix (under UNIX use ce or vi).
Step 5: Copy files from current directory to Cromix/Unix transfer area:
tar -cvf /dev/std? ** (for all files)
Otherwise replace ** with a file name or file list.
Step 6: Sign off Cromix and boot Unix
Step 7: Goto directory where you want the files placed, e.g., /cdos:
cd /cdos
Step 8: Copy files from Cromix/Unix transfer area to current directory:
tar -xvf /dev/std? (for all data)
tar -xvf /dev/std? filename (for one file)
Step 9: Verify files have been copied by doing a directory listing (ls).
Step 10: Become a superuser and set the ownership and access of the files:
su
chown ownername **
chgrp users **
chmod (numbers) **

Note: ownername is the name of who is to own the files, e.g., root. The ambiguous file reference, **, can be changed to a file name or list of file names.

Assignment of Disk Space in Unix

The next example demonstrates how to assign additional disk areas in UNIX. It's in the manuals, but you have to put all the pieces together from different sections. The example is for a 140 meg hard disk with the original disk partitions defined as follows:

std0—Cromix system and data areas
std1—Unix swap area
std2—Unix root area and some data area
std3—Unix data area
std4 thru std6—Open (unformatted) area
std7—Transfer area between Cromix and Unix

References: *Introduction to UNIX System V* manual.

1) See page 7 for the total blocks referenced (40000) with 1 block equal to 1 Kilobyte (page 23). As std2 and std3 are defined they have a total of 40000 Kbytes.

2) It follows that the next free partition is std4. The discussion that follows shows how to make this space available to Unix (a mountable partition).

To open up area std4 the following steps are taken:

Step 1: Log on as root
Step 2: Determine which file format utility to use (makfs or mkfs): compare the disk areas with those listed on page 37; If they match then either command will work. DO NOT use makfs if they don't match
Step 3: Make a file structure on partition std4:
makfs /dev/std4 (–blocks) (disk numbers)
The number of blocks can be found using the Cromix command: diskinfo std31. Disk numbers are given on page 43.
Step 4: The existence of the new disk partition is made known to the system with:
mount /dev/std4 /mnt
cd /mnt
mklost + found
cd /
umount /dev/std4
Step 5: Place std4 in the mount table:
ce /etc/rc
look for line "add mount here" and add the following line:
mount /dev/std4 /mnt
This completes the assignment of std4.

Compiling Programs under Unix

The following is a compilation of information regarding the use of the Fortran compiler and ulinker. It may be of help to others who are using Unix. It is derived from conversations with Cromemco sales personnel and their support staff, from Tom Ronayne — a Cromemco dealer in Detroit, and Bob Baratano from MTU at Houghton.

First, to compile, the command file ftn is used:

```
set -e
/usr/bin/fortran $1.for
/usr/bin/code $1.i
```

To link type:

```
ulinker
```

and answer questions. You can also redirect the answers to ulinker from a file. Always use complete pathnames it avoids many problems.

Example:

```
usr/carl/linkout          ;output file from ulinker
/usr/carl/main.o          ;output file used by the cc
/usr/carl/main.obj        ;main program
/usr/carl/openf.obj       ;routine called by program
/usr/tom/readfile.obj     ;same as openf
```

The lines above are the complete file for ulinker to work on.

Type:

```
ulinker v above__file__name
```

When done type:

```
cc -o /usr/carl/main /usr/carl/main.o
```

When done type:

rm /usr/carl/main.o

The program can then be run with the following command:

/usr/carl/main

Comments or questions regarding these procedures can be directed to:

Carl J. Wick
7082 P Road
Gladstone, MI 49829

Editor's Note:

The following routines were contributed by Randall Pick, who writes:

Setting C-10 Time from Cromix

The Z-80 assembly listing is for a program which can be used to set the time on a C-10 used as a terminal on a Cromix system. The way the program is written an "v ESC v-" is sent to toggle the status line. Since this is a toggle it will also turn the status line off if it is already on, so the user may wish to change the escape sequence at locations 149h and 14Ah to nulls if this is not desired. I wrote the program this way because my watch quit and the C-10 was my only clock. I was playing around with I/O ports and would frequently have to reset the system so this was more convenient for me.

```
;**********************************************************************
;                                                                     *
;   Assembly listing for program to toggle status line and set time on *
;   C-10 computer when used as a terminal under Cromix. The program has *
;   been tested under Cromix 20.63.                                   *
;                                                                     *
;   By Randal W. Pick, Route 2, Box 1125, Sikeston, MO 63801 - 07/03/86 *
;                                                                     *
;**********************************************************************

        ORG 100H        ; .COM files always begin execution at 100h
        JSYS 32H        ; Cromix system call to get time (or use RST 8 32H)
        LD A,E          ; Hours are returned in E register. Save in A
        LD B,H          ; Minutes are returned in H register. Save in B
        LD C,L          ; Seconds are returned in L. Save in C
        LD IX,TIME      ; Pointer to where to store HH:MM:SS
        LD H,0          ; Set dividend high byte to 0 for the divide
        LD L,A          ; Put hours in dividend low byte for divide
        CALL BINDEC     ; Divide & convert to ASCII
        LD (IX+0),L     ; Store hours tens digit
        LD (IX+1),E     ; Store hours ones digit
        LD L, B         ; Put minutes in dividend low byte for divide
        CALL BINDEC     ; Divide & convert to ASCII
        LD (IX+2),L     ; Store minutes tens digit
        LD (IX+3),E     ; Store minutes ones digit
        LD L,C          ; Put seconds in dividend low byte for divide
        CALL BINDEC     ; Divide & convert to ASCII
        LD (IX+4),L     ; Store seconds tens digit
        LD (IX+5),E     ; Store seconds ones digit
        LD DE,LINE      ; Point to beginning of string to send to C-10
        LD C,09H        ; Select print buffered line system call
        CALL 5          ; Print the line
        LD C,0          ; Prepare to terminate with program abort system call
        CALL 5          ; See you next time!
;
BINDEC: PUSH BC         ; Need to save minutes & seconds
        LD DE,0AH       ; Divisor is 10
        LD C,8AH        ; Select divide integer system call
        CALL 5          ; Divide! (quotient in HL, remainder in DE)
        LD A,L          ; Have to use A to add. (only need low byte)
        ADD A,30H       ; Converts binary number to ASCII (if less than 10)
        LD L,A          ; Put ASCII tens digit back where it belongs
        LD A,E          ; Do the same...      /
        ADD A,30H       ; for the ...        /
        LD E,A          ; ones digit.       /
        POP BC          ; Get minutes & seconds back
        RET             ; Return
;
LINE:   DB 1BH,'-',1BH,' ' ; Esc codes to toggle status line & set time on C-10
TIME:   DB 0,0,0,0,0,0,'$' ; Storage area for HH:MM:SS. Buffered line....
                           ; terminates with "$".
```

Reading Cromix Date & Time with SBASIC

I have also included an SBASIC subroutine that reads the date and time from a Cromix system and returns them in string

variables Dat$ and Tim$. When using Rem statements to store machine code in this fashion it should be kept in mind that the initial Rem statements are clobbered when the machine code is Poke.dt213.d in, so if you Save or List the program after it has been run there is no telling what those lines will look like afterward.

```
10   Rem "GETDATE.LST" by Randal W. Pick, Rt. 2, Box 1125, Sikeston, MO, 63801
20   Rem
30   Rem SBASIC subroutine to get date & time from Cromix and return the values
40   Rem in the variables Date$ ("MM/DD/YY") and Time$ ("HH:MM").
50   Rem There MUST be a Rem statement at least 30 characters long at the beginning
60   Rem of the program to store the machine code.
70   Rem Obviously this subroutine can only be used under the Cromix CDOS simulator.
80   Rem The program has been tested under Cromix 20.63.
90   Rem
100  Rem  BASIC DRIVER FOR MACHINE CODE ROUTINE TO RETRIEVE DATE & TIME FROM CROMIX
110  Rem
120  Rem   POP IX OFF STACK TO ALLOW RETURN TO BASIC AND LOCATE DATA
130  Data %00DD%,%00E1%
140  Rem
150  Rem JSYS 30 : CALL .GETDATE CROMIX CALL
160  Data %00CF%,%0030%
170  Rem
180  Rem LD (IX+0),H : STORE MONTH
190  Data %00DD%,%0074%,%0000%
200  Rem
210  Rem  LD (IX+1),L : STORE DAY
220  Data %00DD%,%0075%,%0001%
230  Rem
240  Rem  LD(IX+2),E : REM STORE YEAR
250  Data %00DD%,%0073%,%0002%
260  Rem
270  Rem  JSYS 32 : CALL .GETTIME CROMIX SYSTEM CALL
280  Data %00ÇF%,%0032%
290  Rem
300  Rem  LD (IX+3),E : STORE HOURS
310  Data %00DD%,%0073%,%0003%
320  Rem
330  Rem  LD (IX+4),H : STORE MINUTES
340  Data %00DD%,%0074%,%0004%
350  Rem
360  Rem  RET : RETURN TO BASIC
370  Data %00C9%
380  Rem
390    For Location=%0103% To %0103%+21
400    Read Byte : Poke Location,Byte
410    Next Location
420  A=Usr(%0103%,%0119%)
430  Month=Peek(%0119%) :Day=Peek(%011A%) :Year=Peek(%011B%) :Hour=Peek(%011C%) :Minute=Peek(%011D%)
440  Month$=Str$(Month) : Z$=Month$ : Gosub Datelen : Month$=Z$
450  Day$=Str$(Day) : Z$=Day$ : Gosub Datelen : Day$=Z$
460  Year$=Str$(Year) : Z$=Year$ : Gosub Datelen : Year$=Z$
470  Hour$=Str$(Hour) : Z$=Hour$ : Gosub Datelen : Hour$=Z$
480  Minute$=Str$(Minute) : Z$=Minute$ : Gosub Datelen : Minute$=Z$
490  Dat$="  /  /  " : Tim$="  :  "
500  Dat$(0,-2)=Month$ : Dat$(3,-2)=Day$ : Dat$(6,-2)=Year$
510  Tim$(0,-2)=Hour$ : Tim$(3,-2)=Minute$
520  Return
530 *Datelen
540  If Len(Z$)=2 Then Return
550  Z$(1,-1)=Z$(0,-1) : Z$(0,-1)="0" : Return
```

Continued

Direct Console I/O with SBASIC

A third program (INKEYS$.SUB) is an SBASIC subroutine that reads the keyboard. I have always been a little disappointed that SBASIC did not include a function equivalent to the Inkey$ function of other BASICs that reads the keys as they are pressed. The example given in my Structured Basic manual (page 220, Nov. '82 edition) uses the ports to accomplish this. This routine, however, indicates that the keyboard uses I/O ports 0 and 1 while both the C-10 tech manual and the C-10 user manual show this to be ports 30 and 31 hex. I never could get this routine to work properly, even with the right ports. Besides that, this technique could not be used under the Cromix CDOS simulator.

Editor's Note: For an explanation of and solution to the port polling problem referred to above, see I/O NEWS Vol. IV, No. 6.

I found that a much better way to do this is to use a machine code subroutine that uses the CDOS (and CP/M) direct console I/O system call 06 hex. This subroutine is very short and will also run under Cromix with any type of terminal. The subroutine is also an example of how to use string variables for the storage of machine code subroutines. This is not described in the SBASIC manual and has the advantages that it makes it convenient to use several machine code subroutines in the same program and does not clobber any Rem statements residing in memory.

```
   10  Rem   Subroutine to read the keyboard from SBASIC
   20  Rem   By Randal W. Pick, Rt. 2, Box 1125, Sikeston, MO, 63801   July 8,1986
   30  Rem
   40  Rem   First initialize the string containing the machine code
   50  Rem   with a "GOSUB INIT" command.
   60  Rem   After that every time a "GOSUB INKEY" command is executed
   70  Rem   the subroutine will wait for keyboard input and return
   80  Rem   the ASCII code of the pressed key in the variable "Key".
   90  Rem
  100  Rem   This subroutine should work with any terminal under the Cromix
  110  Rem   CDOS simulator as well as on the C-10 and has been tested under
  120  Rem   Cromix 20.63 and Cromix-Plus 31.05.
  130  Rem
  140  Rem   The machine code executes the following instructions:
  150  Rem
  160  Rem   POP BC
  170  Rem   POP DE
  180  Rem   CALL 5
  190  Rem   LD D,0
  200  Rem   LD E,A
  210  Rem   RET
  220  Rem
50000 *Init : Rem   Load machine code into string variable
50020  Data %00C1%,%00D1%,%00CD%,%0005%,%0000%,%0016%,%0000%,%005F%,%00C9%
50040  Integer I : Dim Code$(8)
50050    For X=0 To 8
50060    Read I : Code$(X,-1)=Chr$(I)
50070    Next X
50080  Var'address=Adr(Code$)
50090  Return
50100 *Inkey
50110  Noesc
50130  Key=Usr(Var'address,%0006%,%00FF%)
50140  If Key=0 Then Goto 50130
50150  Esc : Return
```

The source and compiled assembly programs, c10time.z80 and c10time.com, as well as the Sbasic subroutines, getdate.sub and inkey$.sub, are available from *I/O NEWS*. Call or write the office to request them on 5 inch disk or via modem.

Questions or comments regarding these routines can be directed to:

Randall W. Pick
Sikeston Power Station
P.O. Box 370
Sikeston, MO 63801

Editor's Note:

The following was contributed by Bob Staudenmaier, who writes:

I have received many inquiries for this little handout entitled "Converting the Screen Editor for Non-Cromemco Terminals" after I mentioned it in my article "Another Cromix Screen Editor" (see *I/O NEWS* Vol. IV, No. 1). I do not know the original author of the handout (it may or may not have been Cromemco) so I cannot take credit for writing it. I did, however, make a few changes and added a "whole 'nother page."

Editor's Note: the original handout did originate at Cromemco and was issued as a Technical Bulletin.

Converting the SCREEN Editor for Non-Cromemco Terminals

To use SCREEN.COM or SCREEN.BIN with a NON-CROMEMCO terminal, load the SCREEN file with DEBUG (or PATCH) and substitute in the appropriate ESCAPE or CONTROL codes for your terminal. To compute the address of the terminal functions, add the relative address in the first column to the base address of your version of SCREEN and enter these values in the second column. See the other chart for base addresses.

RELATIVE ADDRESS	ABSOLUTE ADDRESS	BYTES	ASCII	3102 INTERPRETATION
(Codes sent from Screen program to terminal)				
000	[ ]	1B 45	Esc E	Clear Screen
004	[ ]	1B 48	Esc H	Home Cursor
007	[ ]	08	^H	Cursor Left
00A	[ ]	1B 43	Esc C	Cursor Right
00D	[ ]	1B 41	Esc A	Cursor Up
013	[ ]	1B 4B	Esc K	Clear to end of line
016	[ ]	1B 4A	Esc J	Clear to end of page
01A	[ ]	1B 46	Esc F	Cursor addressing
01D	[ ]	1B 5A	Esc Z	Cursor display ON/OFF
020	[ ]	1B 5A	Esc Z	Cursor display ON/OFF
023	[ ]	1B 6C	Esc L	Start Blink
026	[ ]	1B 6D	Esc M	Normal Video
(Information concerning display format)				
029	[ ]	50		Characters per line
02A	[ ]	18		Lines per screen
(Codes sent from Screen program to terminal)				
02C	[ ]	1B 4C	Esc L	Insert Line
02F	[ ]	1B 4D	Esc M	Delete Line
(Codes sent from terminal to Screen program)				
032	[ ]	17	Cntrl-W	Cursor Up
033	[ ]	0B	Cntrl-K	Cursor Up
035	[ ]	0A	Cntrl-J	Cursor Down
036	[ ]	1A	Cntrl-Z	Cursor Down
038	[ ]	01	Cntrl-A	Cursor Left
039	[ ]	08	Cntrl-H	Cursor Left
03B	[ ]	04	Cntrl-D	Cursor Right
03C	[ ]	0C	Cntrl-L	Cursor Right
03E	[ ]	1B	Cntrl-[	Escape
040	[ ]	03	Cntrl-C	
042	[ ]	19	Cntrl-Y	

```
BASE ADDRESSES OF TERMINAL FUNCTION TABLE FOR SCREEN VERSIONS
=============================================================

PROGRAM     VERSION   BASE ADDRESS      BASE ADDRESS
                        (DEBUG)           (PATCH)

SCREEN.COM   1.23       01B0              00B0
SCREEN.BIN   1.23       01B2              00B2
SCREEN.COM   1.45       01B6              00B6
SCREEN.BIN   1.45       01BD              00BD
SCREEN.BIN   1.51       01B6              00B6
```

Notes:

1) If you do not see an entry in the table for your version of SCREEN, you should be able to determine the base address by searching the file with DEBUG (or PATCH). Look for the bytes 1B 54 (the clear screen function in the 3102) usually between 1A0 and 1C0 for most versions.

2) SCREEN assumes that the user terminal accepts the line before the column (with an offset of 20H) for cursor positioning. Any terminal requiring the column before the line, or ASCII line and column numbers cannot be made to work with this method.

3) The terminal functions clear-to-end-of-line, clear-to-end-of-screen, insert-line and delete-line are not necessary for proper operation of the SCREEN editor. Fill in zeroes for these functions if your terminal does not have them. However, all of these functions will significantly improve operation of the editor and should be used IF POSSIBLE.

4) The terminal functions cursor-display-ON/OFF, start-blink, and normal-video are not important. Fill in zeroes in these locations for most terminals.

5) The maximum length of any control-code sequence is two bytes since a zero must end the string.

The screen editor cannot (with use of this chart only) be made to work with following terminals:

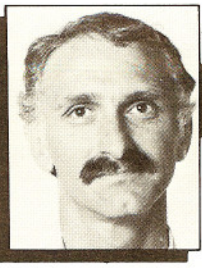
ADDS VIEWPOINT — cursor homes at bottom of screen.

HAZELTINE 1500 — line sent before column for cursor positioning

DEC VT100 — requires ASCII line and column numbers

Anyone discovering information contrary to or excluded from this file should please contact me:

Robert Staudenmaier
325 Ellwood Beach Dr. Apt. 15
Goleta, CA 93117
U.S.A.
(805) 685-6374



C-10 ENCOUNTERS

C-10 ENCOUNTERS is a regular column directed to users of Cromemco's personal computer, the C-10. It is edited by Dr. Tom Beer, of Applied Environmetrics, located at 118 Gordon St., Balwyn, Victoria 3103, Australia. Dr. Beer can be reached by phone during business hours at (03) 817-2571. Submit editorial directly to Dr. Beer.

FAMILY HISTORY & GENEALOGY

I happen to live in a state of Australia called Victoria. Recently my wife, herself a descendent of the McDougals, completed a masters degree in history on Scottish settlers in the western district of Victoria. Now there are lots of wealthy and successful farmers of Scottish descent out there including one, Malcolm Fraser, who was prime minister of Australia in the late nineteen-seventies. However, Jane was not interested in these scions of Scottish success. She wanted to trace the families of 1,412 destitute individuals who had migrated to Australia in 1852 under the auspices of the Highland and Islands Emigration Society. Keeping track of such a large number of unruly Scottish persons required a suitably comprehensive database, and she ended up using dBaseII on the C-10. This worked well, and the capability built into dBase that allows programs to be written allowed her to calculate demographic statistics on her settlers with relative ease.

Even though most of us do not want to keep track of thousands of Scots, all of us have ancestors and many of us have descendents. My congratulations to the Jaenickes who, with the arrival of Sarah Jeanne, have got into the descendent production business. Family history, or keeping track of our relatives, is an area of ever increasing interest — and more and more people are coming to realize that the personal computer can help in this task. What, however, is available for the Cromemco C-10 owner?

Public Domain Genealogy

The first CP/M public domain genealogy package has recently appeared and now forms Volume 8 of the Best of CDOS Public Domain Software (US$25 including postage). It was originally from the OUF (Paris France Osborne Users Group) and written by Sam Washburn, an American in Paris. It has an extensive and well written Wordstar documentation file.

The programs are dBaseII programs so you need to have dBaseII and copy the relevant programs onto the disk. Operation of the program is controlled by a command file so that once dBase is activated, do explorer will get one going. The disk as supplied has a demonstration genealogy of William the Conqueror. I had a bit of trouble using it at first because you either need to know his reference number (which happens to be 1100) or that his real name is actually William de Normandie. From then on it is fine.

The genealogical data base program, EXPLORE, allows you to maintain a data base of people who are related to each other. You can add new people to the database and then link them together by indicating the direct relationships between the people. There are commands in the EXPLORE program that allow you to add and edit the people in your genealogy and their biographies. These commands use certain of the dBASE II functions (edit, browse, append) so you will need to know the control keys that are used in these commands. Otherwise the program is designed for a novice computer user. There are four help-screens. When you first start up it is useful to page through the help-screens as they contain information that is worth knowing right away.

ROOTS/M

Even though the public domain software is cheap, dBase II is not. In addition, dBase is notorious for the slowness of its disk access. Running the public domain genealogy can be a long process. There is, however, a commercial CP/M family tree package called Roots/m in existence (US$69.95 including postage). I ordered it not long after I got my C-10 but could never get it to read the "Cromemco double-sided double-density" disk that the supplier sent. Now that Diskmaster can handle a variety of different disk types I had another try at getting Roots/m to run on the C-10. I got a Kaypro disk and converted it to C-10 format.

Roots/m is a large program that makes extensive use of overlays — by which I mean that it loads in part of the program from the disk as it needs it. As I pointed out in my previous column, the program came with too small a stack for the C-10 and one of my first tasks was to change the address of the stack in order to try and get a working program. For my pains I was rewarded with a program 50% of which worked, but kept returning incorrect error messages for the other 50%. There followed a flurry of correspondence, the end result of which is that I modified the supplied version substantially and got it to run on the C-10.

Roots/m starts up with a menu of choices that includes instructions, calling up the editor, listing the names you have recorded, checking the family of anyone, displaying a pedigree chart (a tree of direct ancestors), printing a pedigree chart or a family group sheet, determining the relationship between any two recorded persons, displaying all births, deaths or marriages on a particular date and finally exiting to the operating system. The package comes with a 98 page manual that describes not only the way that the program works, but also gives a survey of what

genealogy is about and how to go about doing it.

Overall I am impressed with Roots/m. It is efficient in that it executes rapidly and each name of a person or place is stored on disk only once, so that if all your ancestors were born and died in the same place then you will be able to store a very very large number of them on your disk. I have so far entered 50 names and used up only 4K of disk space. There is at least one small bug. I have included an example of a pedigree chart. Position 1 has space for marriage details and the name of a husband or wife, but the program will never print anything there.

The pedigree chart that I have included is of myself and has been configured for the CLQ printer. If you have a 132 column printer then an extra generation can be fitted onto each chart, and the spacing is a lot better. To understand the chart you follow the letter "I" upwards to a father and downwards to a mother. You will thus see that both of the grandfathers of my father were called Adolf. I have sent copies of their pedigree charts to most of my living relatives and hope soon to fill in the blanks.

Roots/m tries a bit too hard to protect the user from himself. For the most part the keyboard is disabled and only a few keys will work. There is a *designation* line (Roots/m name for a status line) on the screen that tells you which key strokes are legal. On most systems this is highlighted and easy to read, but for reasons which I will explain below I do not recommend the use of highlighting (ie. reverse video) on the C-10 so that the designation line fails to stand as a reminder with the clarity that it should have.

I greatly dislike the editor. If you are a perfect individual who never makes a mistake then it is fine. It automatically jumps you from field to field as you fill them in, you need only type "S" when you start a date and it knows that you must have meant September. But woe betide you if you type a lower case "s." It beeps at you. And if you have made a mistake in the place of marriage then it is next to impossible to get the cursor back there to fix it. It can be done, but you have to re-enter a date for the marriage before the editor will permit access to the place-of-marriage field.

Roots/m looks best when the highlighting is enabled. The important messages stand out clearly. Unfortunately, the program seems to assume, at times, that issuing a "turn reverse video command off" at the top left of the screen will remove all highlighting. On the C-10 it does not. The only way that reverse video can be removed is to position the cursor at the exact point where the reverse video starts and issue the normal video code. Because of this peculiarity of the C-10, there are many occasions when there is highlighting in all the wrong places left over from a previous screen. I found it so annoying that I prefer to set up the program with no video attributes at all.

The reason that I can choose, or not choose, video attributes is that there is an installation program supplied which enables you to tailor Roots/m to the peculiarities of your terminal. I had intended to write in detail about the use of this installation program because i) I have been promising to write about terminals ever since my very first column and have yet to get around to doing it and ii) One of my correspondents specifically asked for help in answering the questions to be found in a typical installation program. However, owing to the fact that a pedigree chart takes up a fair few column inches, I will have to defer a discussion of terminal installation to a subsequent column.

Further information on these products is available from Applied Environmetrics.

```
COMMSOFT ROOTS/M    V1.04.01 PEDIGREE CHART                    Chart   1
  2 Oct 1986
                                                  Adolf Beer
                                                  _______________________________
          Imre Beer                              I   8     CONTINUED ON CHART   2
          ______________________________________I
         I  4   BORN      14 Oct 1875            ITerez Beer (nee Kohn)
         I      WHERE     Hungary                I_______________________________
         I      DIED             1956                9     CONTINUED ON CHART   3
         I      WHERE     Hungary
         I      MARRIED  20 Aug 1908
         I
     George (Gyuszi) Beer
     ______________________________
    I  2  BORN      17 Sep 1912
    I     WHERE     Hungary
    I     DIED      27 Jan 1984
    I     WHERE     Australia
    I     MARRIED    8 Jul 1945                   Adolf Kohn
    I                                             _______________________________
    I    IValeria Kohn                           I  10     CONTINUED ON CHART   4
    I    I______________________________________I
    I       5   BORN       3 Jun 1886            ITeresa Beck
    I           WHERE     Hungary                I_______________________________
    I           DIED             1970               11     CONTINUED ON CHART   5
    I           WHERE     Hungary
    I
    I
Tom Beer
______________________________
  1  BORN      27 Jul 1947
     WHERE     Hungary
     DIED
     WHERE
     MARRIED

______________________________
   NAME OF HUSBAND OR WIFE                         Elia Moskovitch Sparber
                                                  _______________________________
    I      Henrik Sparber                        I  12     CONTINUED ON CHART   6
    I      ______________________________________I
    I    I  6   BORN       5 Oct 1881            IKatalin Rottenberg
    I    I      WHERE     Hungary                I_______________________________
    I    I      DIED       5 May 1939               13     CONTINUED ON CHART   7
    I    I      WHERE     Hungary
    I    I      MARRIED  11 Aug 1922
    I    I
    IAnnemarie Sparber
    I______________________________
       3  BORN       7 Jul 1923
          WHERE     Hungary
          DIED
          WHERE
                                                  Arnold Rosner
                                                  _______________________________
         IElizabeth (Erzsebet) Rosner            I  14     CONTINUED ON CHART   8
         I______________________________________I
            7   BORN       6 Jul 1896            IRozalia Roth
                WHERE     now Mukacevi           I_______________________________
                DIED      12 Nov 1966               15     CONTINUED ON CHART   9
                WHERE     Hungary

  No. 1 on this chart is the same person as no.    on chart no.
```

SOFT TOOLS

SOFT TOOLS is a regularly appearing column dedicated to UNIX and Cromix users. Its aim is towards simplifying the administration and maintenance of multi-user systems. It is edited by Tom Ronayne, President of Advanced Programming Techniques Corp. (APTC), P.O. Box 19549, Detroit, MI 48219, (313) 835-0808.

Time and Trouble

Have you ever wondered, after you enter the date and time when you boot either Cromix or Unix, just how the system knows what month and day of the week it is? How about what 30 days from "now" is? Maybe you'd like to know how project management types (I'm one of those) can calculate what date a job is supposed to be finished when all you give 'em is a starting date and number of working days needed to do the job. Maybe you don't care about any of this.

Well, your system keeps track of dates with a value that usually (not always) is the "number of days since sometime." Sometimes, it's the number of days since January 1, 1900; in others it's the number of days since March 1 of some leap year; in still others it's the number of seconds since some date (UNIX does this); in yet others it's some crazy thing that we don't care about here, so we aren't going to bother with them.

This is about "days and dates" tools: Most of the date routines I've seen use some form of a mathematical procedure known as "Zeller's Congruence." When something is congruent, it agrees or coincides with something. For our purposes, congruence means two numbers are congruent with respect to a modulus, or, in English, the remainder after you divide a number by some number — hang in there for a minute, this gets clear later.

The trick is, if you divide the number of days from some base date by seven, you'll have a remainder of zero through six, and, zero through six will represent a day of the week: 0 = Sunday through 6 = Saturday — which is how the system knows what day of the week some date is.

Ok, great. So, how do you get the number of days if you've got a date? And, alternately, how do you get a date when you've got the number of days? What can you do with it? Hang on, we'll get there.

What we're going to do is build some tools that you can use to calculate dates with — they're handy, they're useful, and they're slick. A nice touch is that they'll work on any system, in any language.

What we want to do is keep track of time — not time from the beginning of the universe, but time from some reasonable point that's before any date we're ever going to be concerned with. I use 1964 as that point for reasons that are lost in antiquity — you can use any point you'd like (everything works fine) but—and this is important—you've got to use a leap year, and it must be in the 20th century for the routines presented here (Zeller's Congruence works with the Gregorian calendar for any date after 1582). Remember that leap years are any year equally divisible by 4, and leap centuries are any century year equally divisible by 400 (by the way, the first year of the century is the "1" year, not the "0" year; 2000 is the last year of the 20th century, 2001 is the first year of the 21st century).

These tools cover a 100-year period, more than enough to be useful in our business and personal lifetimes. There are other tools that you can use to cover longer periods — if you're interested, call me and I'll give them to you.

How Do We Give You Tools?

I've got a problem when I try to present tools — I've got to present them in some form that everybody out there can understand. I can do these things in pseudo-code, which looks like English but leaves you with the job of inventing workable code in the language you use, not a real good option. I could present them in Basic, but Basic doesn't work too well for the C, Fortran, and Pascal folks. Also, presenting things in C doesn't work for the Basic folks, and so on around the circle.

Around these parts we work in Fortran, but we don't use Fortran itself if we can help it — we write everything in Rational Fortran (RatFor) — and let the RatFor parser translate rational, structured code into Fortran-77 for compilation. We do this (seemingly) extra step because RatFor lets us write structured code that translates to fast Fortran-77, and, when necessary, we quickly and easily translate structured RatFor code into other languages, like Basic, C, or Pascal. Too, we take great pains to write "vanilla" code, so what we do will run on non-standard compilers (the MicroSoft Fortran compiler, for example, is NOT ANSI-standard Fortran-77 like Cromemco's Fortran-77 compiler is).

I'm going to present routines in RatFor because it's straightforward, easy to read and understand, and easy to translate into a target language you may be familiar with. I believe that highly- structured RatFor/Fortran is easier to understand — if the esoteric stuff is explained — than C is but I'm open to suggestion on other languages or presentation methods.

Before we can start, we need to know a little about RatFor and Fortran-77, so, here's a brief primer.

RatFor was developed by Brian Kernighan and P. J. Plauger — with input from Dennis Ritchie — at Bell Laboratories, and RatFor grammar and syntax is almost identical to C, using Fortran math, data, and format statements. RatFor uses symbols to represent logic, and the symbols are common to Basic, C, and other languages (this makes RatFor a good compromise). Here's a table of comparative symbols:

<table>
<tr><th>RatFor Symbol</th><th>Basic Symbol</th><th>C Symbol</th><th>Means</th></tr>
<tr><td>#</td><td>rem</td><td>/*</td><td>start of comment</td></tr>
<tr><td>none</td><td>none</td><td>*/</td><td>end of comment</td></tr>
<tr><td>|</td><td>or</td><td>||</td><td>logical or</td></tr>
<tr><td>&</td><td>and</td><td>&&</td><td>logical and</td></tr>
<tr><td>^</td><td>not</td><td>!</td><td>logical not</td></tr>
<tr><td><</td><td><</td><td><</td><td>less than</td></tr>
<tr><td><=</td><td><=</td><td><=</td><td>less than or equal to</td></tr>
<tr><td>==</td><td>=</td><td>==</td><td>equal to</td></tr>
<tr><td>^=</td><td>^=</td><td>!=</td><td>not equal to</td></tr>
<tr><td>></td><td>></td><td>></td><td>greater than</td></tr>
<tr><td>>=</td><td>>=</td><td>>=</td><td>greater than or equal to</td></tr>
<tr><td>{</td><td>none</td><td>{</td><td>opening brace</td></tr>
<tr><td>}</td><td>none</td><td>}</td><td>closing brace</td></tr>
</table>

Not too difficult. Fortran-77 formats are pretty obvious: there's an "a" for character types, "i" for integer types, and "f" for floating-point types.

If you use a number with the "a" or "i' format, it defines the width of a printed field; "a10" means 10 characters, "i5" means 5 digits, "f10.3' means 3 decimal places in a field 10 characters wide leaving room for 6 digits before the decimal place.

For example, the Fortran-77 statement

write (*, '(t8, a, i5)') 'This is a string', 10

would print

This is a string 10

on your standard output (the "*" means standard input and output channel, the "t8" means tab to column 8, the "a' means string, the "i5" means integer field with a width of 5). This is equivalent to C's

printf ("\tThis is a string%5d\n", 10);

or Basic's

print tab (8); "This is a string", 10

Again, not too difficult.

You'll notice something about our code when we do a "mixed mode" calculation (where we do mathematical operations on unlike variable types like integer and floating-point) we will ALWAYS include the "ifix," and "float" instructions to "declare the result of the computation." Fortran (and C) will force calculation results: If you declare

integer i
real x, y

and then, in your program have a line

x = i * y

Fortran and C will force the result to be floating-point. You can, though, forget and have "bugs" introduced, so we always use

x = float (i) * y

to assure the result we're expecting. We do the same thing in C.

Well, here goes — this is the first tool, "subdtt," which converts a date like "31jan86" into the number of days since "01mar64." We used a "define" to set the BASE to "64;" which, here, means "1964," a leap year as required. This is written to become a Fortran-77 subroutine; i.e., you "call" it.

It could just as easily be a Fortran-77 function (or a C function) by changing three or four lines.

```
define(BASE,64)
#       subdtt--convert date to elapsed days
        subroutine subdtt (day, month, year, ndays)
#
#       input is a date in the form DDMMMYY
#       output is the number of days since march 1, 19-BASE year
#
#       specifications for calling variables
        character * 3 month
        integer day, year, ndays
#       specifications for functions
        character touper
#       specifications for internal variables
        character * 3 mthnam (12)
        integer daymth (12)
        integer iy, i, m
#       month names
        data mthnam /"JAN", "FEB", "MAR", "APR", "MAY", "JUN",
                "JUL", "AUG", "SEP", "OCT", "NOV", "DEC"/
#       number of days in each month
        data daymth /31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31/
#       routine starts here--set internal year
        iy = year
#       convert the calling month name to upper case
        for (i = 1; i <= 3; i = i + 1)
                month (i : i) = touper (month (i : i))
#       check the calling month name to see if it's any good
        m = 0
        for (i = 1; i <= 12; i = i + 1)
                if (month == mthnam (i))
                        m = i
#       calling month and day of the month any good?
        if (m == 0 | day - daymth (m) > 0 | day <= 0) {
                ndays = -1
                return
                }
#       account for leap years
        if (day - daymth (m) == 0)
                if (m - 2 == 0)
                        if (mod (iy, 4) ^= 0) {
                                ndays = -1
                                return
                                }
#       watch for century rollover
        if (iy <= BASE)
                iy = iy + 100
#       years since 19-BASE year
        iy = iy - BASE
#       months since march
        m = m - 3
#       get correct year and month
        if (m < 0) {
                m = m + 12
                iy = iy - 1
                }
#       calculate days since 01marBASE
        ndays = ifix (float (iy) * 365.25) + ifix (float (m) * 30.6 + 0.5) + day - 1
        return
        end
```

The touper function:

```
define(BIGA,65)
define(LETA,97)
        character function touper (inchar)
#       touper - convert to ascii upper case
        character inchar
#
        if (inchar >= "a" & inchar <= "z")
                touper = char (ichar (inchar) - LETA + BIGA)
        else
                touper = inchar
        return
        end
```

And its inverse, tolowr (we don't use it, but here it is):

```
define(BIGA,65)
define(LETA,97)
        character function tolowr (inchar)
#       tolowr - convert to ascii lower case
        character inchar
#
        if (inchar >= "A" & inchar <= "Z")
                tolowr = char (ichar (inchar) - BIGA + LETA)
        else
                tolowr = inchar
        return
        end
```

Notice that subdtt calls upon another subroutine: touper. In-line code to perform the conversion could have been included within subdtt, but since lower-to-upper case conversion is such a common programming task it becomes worthwhile to make a tool of it.

I've defined touper and tolowr as closely as I can to their "congruent" C functions, toupper and tolower — you can only use

six characters for Fortran-77 variable names.

This is the idea of using Soft Tools: you examine the things you do over and over again — like upper- and lower-case conversions — and see if you've got a candidate for a tool. If you do, make a library routine out of the thing you do over and over again, and you'll drastically cut down your program development time.

If you call subdtt with a date, you'll get back the number of days since March 1, 1964 — from just under 8000 days to just under 8400 days during this year (1986). Nice, but now what?

Well, now you need subttd, which converts the number of days back into a date:

```
define(BASE,64)
#       subttd--convert elapsed days to date
        subroutine subttd (ndays, day, month, year)
#
#       the input is elapsed days since march 1, 19-BASE year
#       output is a date in the form DDMMMYY
#
#       specifications for calling variables
        character * 3 month
        integer ndays, day, year
#       specifications for internal variables
        character * 3 mthnam (12)
        integer n, m
#       month names
        data mthnam /"Jan", "Feb", "Mar", "Apr", "May", "Jun",
                     "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"/
#       routine starts here--is the number of days good
        if (ndays < 0) {
                day = 0
                month = "Xxx"
                year = 0
                return
                }
#       calculate years since 19-BASE year
        year = float (ndays + 1) / 365.25
#       how many days since march 1
        for (;;) {
                n = ndays + 1 - ifix (float (year) * 365.25)
                if (n > 0)
                        break
                year = year - 1
                }
#       calculate month this year (mar = 0, feb = 11)
        m = ifix (float (n) / 30.6)
#       now calculate the day of this month
        for (;;) {
                day = n - ifix (float (m) * 30.6 + 0.5)
                if (day > 0)
                        break
                m = m - 1
                }
#       correct the month (jan = 1, dec = 12)
        m = m + 3
        if (m - 12 > 0) {
                m = m - 12
                year = year + 1
                }
#       set the year for the 19-BASE year offset
        year = year + BASE
        if (m <= 0)
                m = 3
#       move the month name into month
        month = mthnam (m)
#       watch for the century rollover
        if (year > 99)
                year = year - 100
        return
        end
```

Now, the icing — what day of the week is a particular date? Here's subdow for converting the number of days calculation from subdtt to a day of the week:

```
#       subdow--convert elapsed days to day of the week
        subroutine subdow (ndays, dow)
#       specifications for calling variables
        character * 3 dow
        integer ndays
#       specifications for internal variables
        character * 3 day (7)
#       days of the week
        data day /"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"/
#       if ndays equal -1, the date wasn't any good
        if (ndays >= 0)
                dow = day (mod (ndays, 7) + 1)
        else
                dow = "Xxx"
        return
        end
```

To tie everything together — and to show you how to use these routines — here's a little demonstration program. The program — demo — calculates the number of days, converts the number of days back into a date, and calculates the day of the week for the "31st" of every month in 1986. Now, we know that there isn't a "31st" of every month, so we'll see what happens when we try to calculate an invalid date:

```
        program demo
#       make space for the month and day-of-the-week names
        character * 3 month (12), dow (12)
#       make space for the day, year, and number of days
        integer day (12), year (12), ndays (12)
#       initialize days and ndays arrays to zero
        data day /12 * 31/, ndays /12 * 0/
#       initialize month names
        data month /"jan", "feb", "mar", "apr", "may", "jun",
                    "jul", "aug", "sep", "oct", "nov", "dec"/
#       initialize year array
        data year /12 * 86/
#       program starts here
        for (i = 1; i <= 12; i = i + 1) {
#               display the date we're going to convert
                write (*, "(a, i2, a, i2 \)") "Original: ", day (i), month (i), year (i)
#               call subdtt to convert to number of days
                call subdtt (day (i), month (i), year (i), ndays (i))
#               display the results
                write (*, "(5x, a, i4 \)") "Elapsed: ", ndays (i)
#               call subttd to convert back to a date
                call subttd (ndays (i), day (i), month (i), year (i))
#               display the results
                write (*, "(5x, a, i2, a, i2 \)") "Converted: ", day (i), month (i), year
#               call subdow to get the day-of-the-week
                call subdow (ndays (i), dow (i))
#               display the results
                write (*, "(5x, 2 a)") "Day: ", dow (i)
                }
        end
include subdtt.rfr
include subttd.rfr
include subdow.rfr
include touper.rfr
```

Here's what running this demo program produces:

In Demo	From Subdtt	From Subttd	From Subdow
Original: 31jan86	Elapsed: 8006	Converted: 31Jan86	Day: Fri
Original: 31feb86	Elapsed: -1	Converted: 0Xxx 0	Day: Xxx
Original: 31mar86	Elapsed: 8065	Converted: 31Mar86	Day: Mon
Original: 31apr86	Elapsed: -1	Converted: 0Xxx 0	Day: Xxx
Original: 31may86	Elapsed: 8126	Converted: 31May86	Day: Sat
Original: 31jun86	Elapsed: -1	Converted: 0Xxx 0	Day: Xxx
Original: 31jul86	Elapsed: 8187	Converted: 31Jul86	Day: Thu
Original: 31aug86	Elapsed: 8218	Converted: 31Aug86	Day: Sun
Original: 31sep86	Elapsed: -1	Converted: 0Xxx 0	Day: Xxx
Original: 31oct86	Elapsed: 8279	Converted: 31Oct86	Day: Fri
Original: 31nov86	Elapsed: -1	Converted: 0Xxx 0	Day: Xxx
Original: 31dec86	Elapsed: 8340	Converted: 31Dec86	Day: Wed

No great shakes, but note what happens when you try to convert an invalid date: you get -1, 0, and Xxx in appropriate places. The idea of a tool is that it does something — and, if it can't do what you want it to, it ought to let you know. That's why the extensive checking for validity goes on in the routines and why you get a -1 or Xxx to indicate an error.

These routines are written to be completely independent of each other (except for the BASE definition) so they can be included in a library of tools. If you have a Cromemco compiler, you have a library manager. You can compile these routines (in your choice of language) and use the Library program to build ".obj" libraries for inclusion with your programs. As time goes on, we'll build libraries of tools — like these — using the facilities included with Cromemco compilers.

Here, you'd create the library with subdtt, subttd, subdow, touper, and tolowr in order — a tool can "call" another tool (subdtt calls touper).

Practical Application

In my business, I'm given credit by my suppliers and I grant credit to my customers. Frequently, it's "30 Days." Well, I like to know what date 30 days is — either when I have to pay for something or when somebody is supposed to pay me for what they bought. I use these routines to find those dates:

```
#       due--calculate the due date of a bill
        program due
        character * 3 month, dow
        integer day, year, ndays
        write (*, "(a \)") "Enter the Invoice Date (ddmmmyy): "
        read (*, "(i2, a, i2)") day, month, year
        call subdtt (day, month, year, ndays)
        if (ndays == -1)
                stop "Invalid Date."
        call subttd (ndays + 30, day, month, year)
        write (*, "(a, i2, a, i2)") "Due: ", day, month, year
        end
include subdtt.rfr
include subttd.rfr
include touper.rfr
```

Job Duration Calculations

Here's another practical application. Suppose you have a job to do, and you've estimated that it will take you 375 person-hours to do it. You're going to start the job on Friday, January 2, 1987. When will the job be finished, given that you work 8 hours a day, Monday through Friday?

Well, for starters, 375 person-hours is 46.875 working days, which we'll call 47 days. If we worked 47 days straight, we'd just find the number of days for January 2, 1987, add 46 (you subtract 1 because you work "today"), and find out what date the result is. In real life, though, we don't work seven days a week- -so we have to use a "modulus trick" to find out when the 47th working day is complete. Recall that we said that any number of days divided by 7 will have a remainder of 0 through 6, for Sunday through Saturday. If the "mod 7" of your "ndays" is 6, the day is Saturday, and you add two to make it Monday. Here's how to do it:

```
#       job--demonstrate job completion calculation
        program job
        character * 3 month, dow
        integer day, year, ndays, i, durat
#       define initial starting date
        data day /2/, month /"jan"/, year /87/
#       define job duration
        data durat /47/
#       get the number of days and verify the starting date
        call subdtt (day, month, year, ndays)
        if (ndays < 0)
                stop "Invalid date"
#       display the starting day and date
        call subttd (ndays, day, month, year)
        call subdow (ndays, dow)
        write (*, "(2 a, i3, a, i2)") "Start:  ", dow, day, month, year
#       loop for the duration of the job
        for (i = 1; i <= durat - 1; i = i + 1) {
                ndays = ndays + 1                   # add one day
                if (mod (ndays, 7) == 6)            # is this saturday?
                        ndays = ndays + 2           # make it monday
                }
#       ndays now equals the ending date--show day and date
        call subttd (ndays, day, month, year)
        call subdow (ndays, dow)
        write (*, "(2 a, i3, a, i2)") "Finish: ", dow, day, month, year
        end
include subdtt.rfr
include subttd.rfr
include subdow.rfr
include touper.rfr
```

Over the years, I've used these routines in many applications that I've either written for my own or customer's use. Anytime I need to do date and time calculations, these routines are included. If you can include them in your own application library, please do so and feel free to think up as many uses for them as you can.

As always, these routines — both in source (Ratfor, Fortran, and C) and ".bin" form — are available on our Cromix-Plus system: dial in at 2400, 1200, or 300 baud at (313) 835-0809, log in as "guest," and copy the routines to your system.

If you don't have ccall, send us a blank disk (5-1/4" or 8"), and $5.00 and we'll copy them for you. Be sure to tell us what format; i.e., CDOS, Cromix, UNIX, or MS-DOS, and what density and sides. CD

ProCall/PC+

Continued from front cover

these two systems, once irreconcilable, can now be integrated and made to work together. The wizard behind this magic is **ProCall/PC Plus**

ProCall/PC Plus is the most recent descendant in a long line of communications programs offered by Protomatrix Software Development (PSD) which began with ProCall for the C-10 (under CDOS), progressed to ProCall for S-100 CDOS and Cromix, and eventually found its way to MS-DOS as ProCall/PC. As such, it has matured in a way that only the most popular of software can: through continual enhancements and refinements derived from extensive customer feedback. For example, ProCall/PC Plus offers 3102 terminal emulation (as well as DEC's VT52 and VT100) and the Cromix **rfile/sfile file transfer protocol**. These features and capabilities are examined more closely in the remainder of this article.

Documentation

Whenever I review a software package, irregardless of the application, the first thing I look at is the accompanying documentation—it is usually the weak link in the chain. Not so with ProCall/PC Plus. If every software company provided documentation that was as clear, as thorough and read as easily as PSD's, their customer support lines would not be ringing off the hook.

The ProCall/PC Plus Users Manual goes far beyond providing the user with the necessary information to use the program. It's educational in a much more general sense, with many examples given, interesting historical notes regarding telecommunications, and useful tips provided by other users. It also offers something even more rare in technical writing of this nature—humor—one example being the inclusion in the glossary of the word "defenestration," which is "the act of throwing something through, from, or out of a window." Actually, it is the best Users Manual I have ever read (and I say that a little grudgingly, having written a few Users Manuals myself).

Installation

Included with the ProCall/PC Plus Users Manual is a separate sheet of instructions for installing the software. As it happens, this is a straightforward affair handled for the most part by the "install" command file present on the distribution diskette.

PSD now employs a copy protection scheme which is transparent to the user, but which does prevent unauthorized duplication of the program. It is more understanding and forgiving than many schemes in that it allows two (2) installations to be made. In fact, the number of backups which can be made is unlimited—there is an "uninstall" routine which takes the program from the backup (hard or floppy disk) and reinstates it on the original, resetting the installations counter along the way—something you would want to do if your working floppy disk backup was beginning to show signs of wear.

Ease of Use

Like all of its earlier PSD predecessors, ProCall/PC Plus is extremely easy to use. To illustrate, I'll relate my first experience with it.

Since I don't have a PC (yet), my friend Bob let me use his (a Leading Edge Model D with hard disk running MS-DOS version 3.1 with a Hayes Smartmodem 1200) in order to do this review. After we installed the program, I reached for the Users Manual (which I had not yet looked at), but before I could open it Bob had called up ProCall and was moving through the set of communications settings (baud rate, parity, etc.) that were neatly displayed on the screen (see Figure 1). The next thing I knew he had the Main Menu displayed (see Figure 2), and after a moments perusal, had selected the "M" (Modem) command to dial my CS-100's number. Once connected I intervened, remembering the "W" (Wake-Up) command which sends a carefully timed sequence of Carriage Returns. A few seconds later we were greeted with my system's Login prompt. After logging on we experimented with some of ProCall/PC Plus' unique features: the 3102 terminal emulation and some file transfers using rfile and sfile. We did this without ever having to resort to the manual for instructions. Now that's user-friendly!

It was a real thrill to call up WriteMaster, edit a file, convert it to ASCII format, spell-proof it with SpellMaster, and to do all this seated in

Figure 1

Communications Settings

Action:	**Call Mode**	Answer Mode	Disconnect Call		
Speed:	110	150	**300**	1200	
	2400	4800	9600	19200	38400
Stop Bits:	**1**	2			
Parity:	**None**	Odd	Even	Mark	Space
Word Length:	7	**8**			

[↑] [↓] and [←] [→] change settings. [RETURN] selects, [ESC] exits

Figure 2

ProCall/PC Communications System Command Menu

ProCall/PC System Commands

[D]	Disk and Directory Functions	[Q]	Query/Change ProCall Modes
[I]	Communication Port Settings	[J]	Jump to DOS - Run a Program

ASCII Receive Data Commands

[C]	Capture ASCII Text Mode ON	[F]	File Captured Text onto disk
[X]	Capture ASCII Text Mode Off	[Z]	Zap (Clear) Capture Buffer
[V]	View Text in Capture Buffer	[P]	Print Session (ON/OFF)

ASCII Data Sending Commands

[G]	Get File into SEND Buffer	[O]	Send Text in SEND Buffer
[U]	View Text in SEND Buffer	[S]	Send Sign-On Message

ASCII/BINARY file transfer and other commands

[T]	Transfer File using Protocol	[N]	Specify Nulls sent after C/R
[W]	Wake-up Rmt. (send [RETURN]s)	[M]	Modem - Dial a Phone Number
[B]	Send BREAK	[L]	List Macro Definitions
[R]	Replace current Profile	[E]	Terminal Emulation Menu*
[ESC]	Exit this menu	[A]	Abort and EXIT ProCall/PC

Request =>

front of a PC clone. To then transfer the file from the Cromix system to the PC (using the error correcting rfile/sfile protocol) and print it out on the printer was nothing short of magic.

The only hindrance I experienced in working with WriteMaster was getting accustomed to the PC's function key definitions. All of the necessary 3102 function keys are accounted for, but since there are only ten keys it is necessary to use the SHIFT, CONTROL, and ALT keys to access many of them. The function key translations are provided in the manual in a tabular format, which is a little awkward to read. The solution, however, is simple: make a template to fit over the function keys.

Special Features and Capabilities

After reading the Users Manual I learned that Bob and I had really gone about it all wrong, despite the ease with which we accomplished what we did. In fact, setting up the communications, dialing the phone number, going through the login procedure, and transferring the file could have been accomplished with a single keystroke or totally automated.

This is because ProCall/PC Plus has a number of flexible and powerful capabilities which allow the user to configure the program to any given set of circumstances. One option, a command line processor, allows the user to specify many of the details for the communications session when ProCall is called from the command line. For example, with the following command (which appears on two lines but would be typed on one):

```
ProCall SFP myprog *555-1212 -m
?:name\?:pswd\$ -e ex\
```

ProCall/PC would enter file transfer mode (the S option) at 1200 baud (the F option—for FAST). The modem would dial the number specified after the asterisk. Once the remote answered, the log-on string (following the -M code) would be sent to log on: ProCall would wait until it received a ":" (the ?:, which it would with the Cromix Login: prompt); when it did it would send "name" and wait for another ":", which it would get with the prompt for Password:, at which point it would send "paswd." With the logon completed, ProCall/PC would attempt to transfer the file myprog to the remote using PcXfer (the P in the SFP command). The remote must be a system with PcXfer installed. With the transfer completed, the exit macro (following the -e) would be sent to log off the remote and ProCall/PC would return to DOS.

Another alternative is the use of "profiles." A profile is a text file that contains the necessary information for ProCall/PC to conduct a communications session. An example profile is shown in Figure 3.

The user can also define six of the ten function keys available on the PC keyboard. These user-defined function keys can contain macros which could, for example, enable a complex logon procedure to be accomplished with one keystroke. The documentation is very clear and very thorough in describing these capabilities, and offers many examples, one of which details the procedure for automating the rather complex dialog that occurs when one attempts to access The Source through Tymnet.

Another nice feature is that with ProCall/PC Plus the PC can be directly connected to the Cromix system, without the a modem. This is accomplished by connecting a regular RS-232 type cable from the serial port on the PC (usually used for connecting the modem) to one of the serial ports on the Cromix system (from a Tuart, Quadart, or Octart). In this configuration, serial communication can take place at 9600 baud. And with the 3102 terminal emulation enabled, the PC becomes, for all intents and purposes, an intelligent terminal to the Cromix system.

There are many other features which limited space prevents me from detailing: special dialing features, character translation tables for the keyboard, crt, printer, and RS-232 serial interface, and printer support, to name but a few. They're all available to make ProCall/PC Plus one of the most complete and powerful modem communication packages available.

Figure 3

```
; FORMAT:  PROCALL/F FILENAME
; THIS IS A SAMPLE PROFILE FOR PROCALL/PC AND PROCALL/PC plus, VERSION 2.00
; COMMUNICATION SETTINGS
BAUD RATE=9600                      ; 110,150,300,1200,2400,4800,9600,19200,38400,ASK
STOP BITS=1                         ; 1, 2, ASK
PARITY=NO                           ; NONE, ODD, EVEN, MARK, SPACE, ASK
DATA BITS=8                         ; 8, 7, ASK
; DEVICE SETTINGS
COMM PORT=1                         ; 1, 2
PRINTER PORT=1                      ; 1, 2, 3
LINK=MODEM                          ; TYPE OF CONNECTION i.e. MODEM OR DIRECT WIRED
; ADDITIONAL OPTIONS
DUPLEX=FULL                         ; HALF, FULL
RECEIVE LF=OFF                      ; OFF, ON
SEND LF=OFF                         ; OFF, ON
HOST CONTROL=OFF                    ; OFF, ON
AUTO SIGN-ON=OFF                    ; OFF, ON
LOWER CASE=ON                       ; OFF, ON
PRINTER=OFF                         ; OFF, ON
CAPTURE MODE=BUFFER                 ; BUFFER, DIRECT
CAPTURE=OFF                         ; OFF, ON
TERMINAL EMULATION=NONE             ; 3102, VT100, VT52, NONE
NULLS NEEDED BY REMOTE=0            ; 0 - 9 NULLS TO BE SENT AFTER EACH RETURN
; MACRO DEFINITIONS
MACRO1=                             ; (40 CHARS IN ''. FOR NONE, LEAVE BLANK)
MACRO2=                             ;  "
MACRO3=                             ;  "
MACRO4=                             ;  "
MACRO5=                             ;  "
MACRO6=                             ;  "
MACRO7 (AUTO SIGN-ON/ANSWERBACK)=
LOG-ON MACRO=                       ; (60 CHARS IN '' FOR NONE, LEAVE BLANK)
EXIT MACRO=                         ;  "
; OPTIONAL CHARACTARISTICS
MODE=CALL                           ; CALL (NORMAL) OR ANSWER MODE
MODEM INIT='ATS0=2'                 ; STRING TO INIT MODEM FOR ANSWER MODE
PASSWORD=                           ; OPTIONAL 8 CHARATER PASSWORD IN QUOTES
GREETING (BEFORE PASSWORD)=         ; OPTIONAL 1 LINE GREETING IN QUOTES
GREETING (AFTER PASSWORD)=          ; OPTIONAL 1 LINE INSTRUCTIONS IN QUOTES
; DIALING INFORMATION
MODEM=HAYES                         ; HAYES, RACAL, CROMEMCO/VEN-TEL
DIAL PREFIX=                        ; (7 CHARS IN ''. SENT BEFORE NUMBER FOR PBX)
DIAL=                               ; NUMBER TO DIAL WHEN ONLINE
```

Making it easy to establish a connection with another computer is only part of the story. Once connected, the most common thing to do is either upload or download files from one system to the other. Here again ProCall/PC Plus shines, for in addition to the common ASCII file transfers, four other error-correcting file transfer protocols are available: Cromemco's rfile/sfile (Cromix systems), **Kermit** (from Columbia University and used on many UNIX systems and mainframes), **Xmodem** (Ward Christiansen— common among dial-up BBSs), and PSD's own **PcXfer**. With these protocols available it becomes possible to upload and download from almost any system. The documentation is ripe with examples for doing so.

Conclusions

The 3102 terminal emulation along with the rfile/sfile and Kermit file transfer capabilities and ProCall/PC Plus' facility for direct connecting (without a modem) offers Cromemco users a host of exciting possibilities. For one, your PC can function as a terminal on your multi-user Cromix system with full access to Cromemco's 3102-dependent software, such as WriteMaster and SCREEN.

In this configuration, the Cromix system can also act as a file-server for various PCs connected to it, with the

added security offered by Cromix (password and file access). And as a side benefit, it provides a convenient solution to the disk format incompatibilities which exist between Cromemco and MS-DOS.

All of this, combined with PSD's well-deserved reputation for quality and support, makes ProCall/PC Plus an irresistible choice as a communications system. So irresistible that I'm going to have to get me a PC (clone that is) so I can take advantage of it.

The price for ProCall/PC Plus is $249. ProCall/PC sells for $149. For further information contact:

Protomatrix Software Development
1145 Park Heights Drive
Milpitas, CA 95035-4605
(408) 263-8665

Additional Reading:

"ProCall Modem Communications System," Greg Pepper, *I/O NEWS* Vol. IV, No. 3, July/August 1984.

"ProCall/PC Plus—A User's Perspective," Ronn Blaylock, *I/O NEWS* Vol. V, No. 2, January/February 1986.

System 220

Continued from front cover

coprocessors operating at 16 MHz, 16Kb of high speed cache, and the UNIX V.2 Operating System. It is a militarized supermicrocomputer designed with a special ruggedized chassis for operation in a hostile environment (extreme temperature/humidity). These modifications include a 500W power supply (the same as in the CS400), three times the cooling capacity of a System 200, and reinforced mechanical stress points for peripherals internal to the chassis.

The MPS software provides a user-friendly, interactive environment in which the pilot plays an active role. He provides pertinent mission data such as base origin, destination, fuel capacities (extra tanks), weapons to carry, and preferred flight altitude (50, 100, 150 feet — whichever the pilot feels most comfortable with).

The MPS then conducts a "penetration analysis" which draws additional information from a very large database of potential threats. Contained within this database is information regarding the type of threat, its location, its effective range, and geographical characteristics of the site derived from satellite imagery. The computer figures in the terrain features and generates a "mask" which indicates those locations where the aircraft is effectively out of range of any threats. This technique is referred to as "terrain masking."

Within this mask the computer selects the optimal route to the target and back—that being the least lethal and shortest route.

This optimal route is then overlayed on a topographically contoured map which is 3-dimensionally represented on a graphics monitor. The pilot can examine it and make modifications to it. When it is to his liking, he'll copy it onto the "strip" map worn on his leg during the flight. The essential flight plan information — in terms of compass bearings, speeds, altitudes, times, and turning points — is then recorded in the DTC which the pilot takes aboard.

Future enhancements to the system will include a graphics plotter to generate the strip maps used by the pilot, removable hard disk drive capability for added security, more RAM memory, as well as diagnostics specially designed for use by non-technical personnel.

This Mission Planning System represents a supermicrocomputer application at the leading edge of today's technology. Development began in 1984 and continues today, as the same technology is applicable to a wide range of aircraft and other aspects of military flight planning. The system was contracted by the Fort Worth Division of General Dynamics, who awarded the first portion of the multi-part contract to Computer Crossroads of America, Inc. to supply the Cromemco System 220s. This installment of the contract is valued in excess of $3.5 million.

For a profile of Computer Crossroads of America, Cromemco's largest VAR (Value Added Reseller), see the inset on the left.

We're Interested!

We're Really Interested...in what you have to say. Especially about how you use your system...the problems encountered and the solutions effected...unusual uses or environments...and any practical applications you would be willing to share with fellow members. These can be short notes for departments like 'bits & bytes...' and 'Tec Tips,' or full feature articles.

Contact Bill Jaenicke at I/O News for editorial guidelines or assistance. We're interested in unleashing your literary talents.

CHANGE OF ADDRESS

Membership # ____________
Type or Print

Your Name ____________

New Address

____

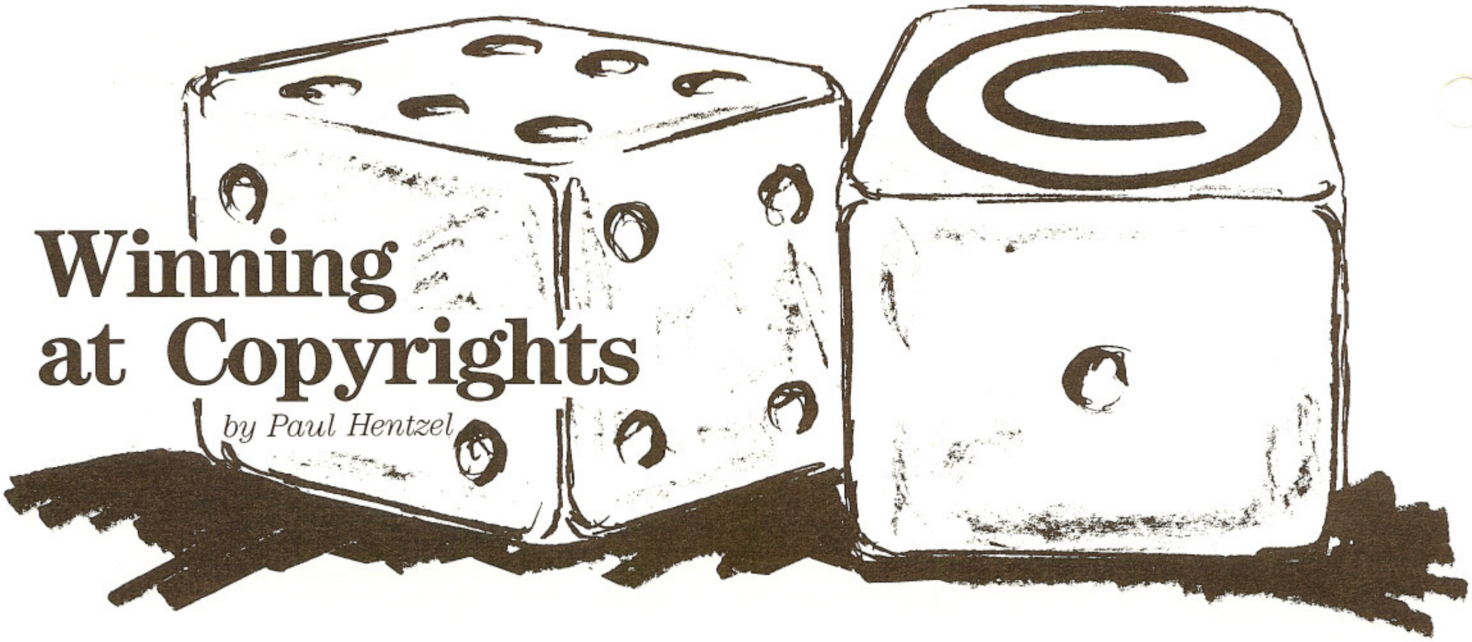
____

New area code
& phone No. (   ) ____________

Sign Here ____________

Date new address in effect ____________

Send address changes to I/O News,



Winning at Copyrights

by Paul Hentzel

Protect Your Software

Developing software is a high risk business; and like any chance situation, you can improve your odds of winning by employing a strategy. I propose the following two-point (minimal cost) strategy to help you win at the "Copyright Table":

1) Always place a copyright notice on every copy or version of your program, from the time of creation (no cost).

2) Register your software with the Copyright Office simultaneously with your first publication or commercial use ($10 filing fee).

You will normally not require the services of a copyright attorney to file and obtain copyright registrations on your software. Preparing a copyright application in Class TX is a simple matter which you can easily do in-house. Failure to protect your software through proper copyright notice may cause the program to become public domain material. Failure to obtain a registration promptly may result in the loss of an extremely favorable set of legal presumptions which arise on behalf of the creator (you) during copyright licensing and litigation. These presumptions are so devastating against copyright infringers (and pirates) that actual copyright litigation rarely occurs.

Following the above two rules does not insure that you will win at copyrights; but it does avoid a heartbreaking loss due to a legal technicality (a TKO).

This is the first in a series of brief articles which will assist you in obtaining copyright protection of your software as a "literary work" in Class TX.

Creation

A copyrightable work such as software is created and becomes eligible for federal copyright protection under the new 1976 Copyright Act (and 1980 amendment) as soon as the work is provided in a tangible and reproducible medium. This first "creation of a work" may occur when the creator (you) writes the program down on paper, or keys the program into permanent memory (magnetic disk or tape). A particularly significant creation occurs as the first typed draft of the program comes out of the printer. Paper copy is the medium in which works are usually handed out to others (published). Each printed copy will require a copyright notice (Rule 1) similar to my notice for this article: "Copyright 1987 Hentzel."

Your notice must include the word "copyright" or the circle C symbol, ©, the year of the first publication, and the name of the copyright claimant (owner).

Before you create the initial tangible copy, your work is merely a notion in your mind, and perhaps the subject of a discussion. These transitory events are neither tangible nor reproducible, and do not qualify for a copyright until you advance them to a more permanent form.

Rights

By merely placing a copyright notice on your software you preserve the following major copyrights:

1) The exclusive right to make copies in any form, hard copies, permanent memory copies, and even temporary memory copies.

2) The exclusive right to prepare derivative works such as improved versions.

3) The exclusive right to distribute the copies by public sale, rental, lease, or other transfer.

Help

Several excellent self-help books are available concerning software protection:

How to Copyright Software. Salone and Elias, 1984.

Legal Care for your Software. Remer, Daniel, 1982.

Both books are from Nolo Press, 959 Parker Street, Berkeley, CA 94710, (415) 549-1976.

The Copyright Office has a "Hotline" phone number, (202) 287-9100, which you may call any time day or night for information. Also the Copyright Office maintains a series of free up-to-date circulars which you may obtain by writing to:

Copyright Office
Library of Congress
Washington, D.C. 20559

Ask for a set of their series R2 circulars which contains lists of all of their other circulars.

If you have any questions or suggestions for future articles you may contact me directly at:

Paul Hentzel
Patent Trademarks Copyrights
441 Nevada Avenue
Palo Alto CA 94301
(415) 326-8254 9am-midnight.

Editor's Note: Paul Hentzel *has been advising Cromemco, Inc. on copyright matters since 1976. The next installment will deal with the particulars of placing the copyright notice on software.*

LOCAL USER GROUPS

Assisting in the formation of Local Cromemco User Groups is one of the services performed by The IACU. We will be happy to provide you with a list of our members in your area, and recommend other contacts to help you organize and maintain a Cromemco computer users group. Just call or write I/O NEWS.

Arizona Association of Cromemco Users
Contact: Jo Ann Drake, President
2207 West Eugie Avenue
Phoenix, AZ 85029
(602) 993-9589

Australia User's Group*
Contact: Minicomp
Minicomp Building
104 Mount Street
North Sydney, NSW 2060
Australia
(02) 957-6800
Meets Monthly
*Publishes "Minicomp/Cromemco" a monthly newsletter

Bay Area Cromemco Users & Programmers (BACUP)
Contact: Raymond Barglow or Alan Walworth
United Word & Data Processing
2345 Fulton Street
Berkeley, CA 94704
(415) 841-0708 or (415) 548-2692

Cromemcohorts
Contact: Dr. Brent Lowensohn
4747 Sunset Blvd.
Los Angeles, CA 90027
(213) 667-8972

Cromemco Users' Group of Australia*
Contact: Tony Stringer
52 Beechwood Avenue
Greystanes, 2145
*Publishes a magazine "CROME-SOMA"

Cromemco Users' Group Holland (CUGH)
Contact: Joop Kohler, Secretary
P.O. Box 120
2910 AC Kieuverkerk a/d IJssel
The Netherlands 01803-13300

Cromemco Users' Group*
Contact: Peter Norman
The University of Newcastle Upon Tyne
Department of Chemical Engineering
Merz Court, Claremont Road
Newcastle Upon Tyne NE1 7RU
England
Newcastle 28511, Ext. 3278
*Publishes Cromemco Users' Newsletter (CUG)

Cromemco Users' Group Ontario, Canada
Contact: Lloyd Parker
Hiram Walker Resources, Ltd.
Suite 600
1 First Cnandian Place
Toronto, Ontario
Canada M5X 1A9
(416) 864-3349

Cromemco Users of Orange County, California
Contact: Michael Peterson
Accountability Systems
700 South Tustin Avenue
Suite B
Orange, CA 92667
(714) 639-4570
Meets third Tuesday monthly

Insystems Pty. Ltd.*
Contact: Norman Rosenbaum
337 Moray Street
South Melbourne, Victoria
3205 Australia
(03) 690-2899, Telex: AA30458
*Publishes "Cromemco UPDATE"
a bi-monthly newsletter

Illinois Users' Group
Contact: Jim Knowles
P.O. Box 631
Elgin, IL 60120
(312) 695-7775

Indonesian Cromemco Users' Group (ICUG)*
Contact: Zafir M.A. Pontoh
Computation Lab
Department of Regional & City Planning
Bandung Institute of Technology
10 Ganesha
Bandung, Indonesia
(022) 82051 ext. 360
*Publishes "BERKALA ICUG"
a monthly newsletter

Microcomputer Users' Group
Contact: Noble Bright
P.O. Box 1
Cape May, NJ 08204
(609) 884-2222
(609) 429-3838
Meets fourth Wednesday monthly

Northwest Association of Cromemco Users (NWACU)
Contact: Jim Illman
403 S. Brandon
Seattle, WA 98108
(206) 763-2099

North San Diego County Users' Group
Contact: Charles Mackey
P.O. Box 397
Fallbrook, CA 92028
(619) 728-6116
Located 30 mi. east of Oceanside

North Texas Cromemco Commercial Users' Group
Contact: Jerrell Johnson
1131 Winterwood
Lewisville, TX 75067
(214) 221-1437
Or call Rocky Hall
at (214) 398-1595
Meets first Wednesday bi-monthly

NY, NY Users' Group
Contact: Charles Perrella
45F Route 303
Valley Cottage, NY 10989
(914) 268-5137

SaCromemco Users
Contact: Alan Whitman
Box 244
Rancho Cordova, CA 95670
(916) 635-6070

Silicon Valley Cromemco Users
Contact: Allan O'Neill
(415) 969-3854 or Emily Ott (415) 854-5818
meeting place provided by:
MCM Enterprises
215 Hamilton Avenue
Palo Alto, CA 94301
Meets fourth Tuesday monthly

W.A. Cromemco Users' Group
Contact: Rae Canning
c/o The W.A. School of Computing
2/294, Rokeby Road
Subiaco, Western Australia 6008

West Germany Users' Group
Contact: Glynnis Long
Tesco GmbH
P.O. Box 10
8714 Weisentheid
West Germany
09383-1237
Total fluency in English & German

Wisconsin Cromemco Users' Group
Contact: Bob Ungemach
6249 West Browndeer Road
Browndeer, WI 53223
(414) 355-1451

The International Association of Cromemco Users

Software Resource Survey

— PURPOSE —

The purpose of this survey is to further expand the list of known software applications running on Cromemco systems. By taking the time to complete and return the questionaire you will be doing yourself and Cromemco users worldwide a tremendous favor. Everyone needs software, but finding the package you need can be terribly frustrating. This survey could eliminate that frustration. So please, help us help you by returning the completed questionaire at your earliest convenience.

Thank you!

— INSTRUCTIONS —

There are three (3) sections to this survey. Put your name and address information in the USER section. Use the SYSTEM section to describe the Cromemco system on which the software is run. If you have different systems running different operating systems and software, photocopy the survey sheet and submit one for each system.

The survey sheet has space for four (4) software package descriptions. If you have more than that, make some additional photocopies. For each package, provide the following information: the **product name**, a brief **description** of the software (spreadsheet, wordprocessor, DBMS, etc.); if you had **problems** in getting the package up and running, so indicate with comment; indicate whether or not you are satisfied with the package. Also, please provide the address and telephone information about the software **vendor** or manufacturer so that we can submit more detailed Product Fact Sheets to them. If the application was designed in-house, please make note of it and indicate if you're interested in having the application included in the **IACU Software Resource Guide**. If so, we'll send you some Product Fact Sheets.

When completed, please mail the survey sheets to:

The I.A.C.U.

USER

Name ____________________ Member No._______

Address ____________________

____

City __________ State _____ Zip _____ Phone __________

Country ____________________

SYSTEM

Model: ____________________

Memory: ____________________

Disks: ____________________

Operating System: ☐ CP/M ☐ CDOS ☐ Cromix

☐ Cromix-D ☐ Cromix-Plus ☐ UNIX ☐ Other________

SOFTWARE		VENDOR INFO
Product Name:	__________	__________
Description:	__________	__________
Problem? ☐ Yes ☐ No	__________	__________
Like it? ☐ Yes ☐ No	__________	__________
Product Name:	__________	__________
Description:	__________	__________
Problem? ☐ Yes ☐ No	__________	__________
Like it? ☐ Yes ☐ No	__________	__________
Product Name:	__________	__________
Description:	__________	__________
Problem? ☐ Yes ☐ No	__________	__________
Like it? ☐ Yes ☐ No	__________	__________
Product Name:	__________	__________
Description:	__________	__________
Problem? ☐ Yes ☐ No	__________	__________
Like it? ☐ Yes ☐ No	__________	__________

LOCAL USER GROUPS

Assisting in the formation of Local Cromemco User Groups is one of the services performed by The IACU. We will be happy to provide you with a list of our members in your area, and recommend other contacts to help you organize and maintain a Cromemco computer users group. Just call or write I/O NEWS.

Arizona Association of Cromemco Users
Contact: Jo Ann Drake, President
2207 West Eugie Avenue
Phoenix, AZ 85029
(602) 993-9589

Australia User's Group*
Contact: Minicomp
Minicomp Building
104 Mount Street
North Sydney, NSW 2060
Australia
(02) 957-6800
Meets Monthly
*Publishes "Minicomp/Cromemco" a monthly newsletter

Bay Area Cromemco Users & Programmers (BACUP)
Contact: Raymond Barglow or Alan Walworth
United Word & Data Processing
2345 Fulton Street
Berkeley, CA 94704
(415) 841-0708 or (415) 548-2692

Cromemcohorts
Contact: Dr. Brent Lowensohn
4747 Sunset Blvd.
Los Angeles, CA 90027
(213) 667-8972

Cromemco Users' Group of Australia*
Contact: Tony Stringer
52 Beechwood Avenue
Greystanes, 2145
*Publishes a magazine "CROME-SOMA"

Cromemco Users' Group Holland (CUGH)
Contact: Joop Kohler, Secretary
P.O. Box 120
2910 AC Kieuverkerk a/d IJssel
The Netherlands 01803-13300

Cromemco Users' Group*
Contact: Peter Norman
The University of Newcastle Upon Tyne
Department of Chemical Engineering
Merz Court, Claremont Road
Newcastle Upon Tyne NE1 7RU
England
Newcastle 28511, Ext. 3278
*Publishes Cromemco Users' Newsletter (CUG)

Cromemco Users' Group Ontario, Canada
Contact: Lloyd Parker
Hiram Walker Resources, Ltd.
Suite 600
1 First Cnandian Place
Toronto, Ontario
Canada M5X 1A9
(416) 864-3349

Cromemco Users of Orange County, California
Contact: Michael Peterson
Accountability Systems
700 South Tustin Avenue
Suite B
Orange, CA 92667
(714) 639-4570
Meets third Tuesday monthly

Insystems Pty. Ltd.*
Contact: Norman Rosenbaum
337 Moray Street
South Melbourne, Victoria
3205 Australia
(03) 690-2899, Telex: AA30458
*Publishes "Cromemco UPDATE"
a bi-monthly newsletter

Illinois Users' Group
Contact: Jim Knowles
P.O. Box 631
Elgin, IL 60120
(312) 695-7775

Indonesian Cromemco Users' Group (ICUG)*
Contact: Zafir M.A. Pontoh
Computation Lab
Department of Regional & City Planning
Bandung Institute of Technology
10 Ganesha
Bandung, Indonesia
(022) 82051 ext. 360
*Publishes "BERKALA ICUG"
a monthly newsletter

Microcomputer Users' Group
Contact: Noble Bright
P.O. Box 1
Cape May, NJ 08204
(609) 884-2222
(609) 429-3838
Meets fourth Wednesday monthly

Northwest Association of Cromemco Users (NWACU)
Contact: Jim Illman
403 S. Brandon
Seattle, WA 98108
(206) 763-2099

North San Diego County Users' Group
Contact: Charles Mackey
P.O. Box 397
Fallbrook, CA 92028
(619) 728-6116
Located 30 mi. east of Oceanside

North Texas Cromemco Commercial Users' Group
Contact: Jerrell Johnson
1131 Winterwood
Lewisville, TX 75067
(214) 221-1437
Or call Rocky Hall
at (214) 398-1595
Meets first Wednesday bi-monthly

NY, NY Users' Group
Contact: Charles Perrella
45F Route 303
Valley Cottage, NY 10989
(914) 268-5137

SaCromemco Users
Contact: Alan Whitman
Box 244
Rancho Cordova, CA 95670
(916) 635-6070

Silicon Valley Cromemco Users
Contact: Allan O'Neill
(415) 969-3854 or Emily Ott (415) 854-5818
meeting place provided by:
MCM Enterprises
215 Hamilton Avenue
Palo Alto, CA 94301
Meets fourth Tuesday monthly

W.A. Cromemco Users' Group
Contact: Rae Canning
c/o The W.A. School of Computing
2/294, Rokeby Road
Subiaco, Western Australia 6008

West Germany Users' Group
Contact: Glynnis Long
Tesco GmbH
P.O. Box 10
8714 Weisentheid
West Germany
09383-1237
Total fluency in English & German

Wisconsin Cromemco Users' Group
Contact: Bob Ungemach
6249 West Browndeer Road
Browndeer, WI 53223
(414) 355-1451

The International Association of Cromemco Users

Software Resource Survey

— PURPOSE —

The purpose of this survey is to further expand the list of known software applications running on Cromemco systems. By taking the time to complete and return the questionaire you will be doing yourself and Cromemco users worldwide a tremendous favor. Everyone needs software, but finding the package you need can be terribly frustrating. This survey could eliminate that frustration. So please, help us help you by returning the completed questionaire at your earliest convenience.

Thank you!

— INSTRUCTIONS —

There are three (3) sections to this survey. Put your name and address information in the USER section. Use the SYSTEM section to describe the Cromemco system on which the software is run. If you have different systems running different operating systems and software, photocopy the survey sheet and submit one for each system.

The survey sheet has space for four (4) software package descriptions. If you have more than that, make some additional photocopies. For each package, provide the following information: the **product name**, a brief **description** of the software (spreadsheet, wordprocessor, DBMS, etc.); if you had **problems** in getting the package up and running, so indicate with comment; indicate whether or not you are satisfied with the package. Also, please provide the address and telephone information about the software **vendor** or manufacturer so that we can submit more detailed Product Fact Sheets to them. If the application was designed in-house, please make note of it and indicate if you're interested in having the application included in the **IACU Software Resource Guide**. If so, we'll send you some Product Fact Sheets.

When completed, please mail the survey sheets to:

The I.A.C.U.

USER

Name ______________________ Member No. ________

Address ______________________

____

City ____________ State ______ Zip ______ Phone ____________

Country ______________________

SYSTEM

Model: ______________________

Memory: ______________________

Disks: ______________________

Operating System: ☐ CP/M ☐ CDOS ☐ Cromix

☐ Cromix-D ☐ Cromix-Plus ☐ UNIX ☐ Other________

SOFTWARE / VENDOR INFO

Product Name: ______________________ ______________________

Description: ______________________ ______________________

Problem? ☐ Yes ☐ No ______________________ ______________________

Like it? ☐ Yes ☐ No ______________________ ______________________

Product Name: ______________________ ______________________

Description: ______________________ ______________________

Problem? ☐ Yes ☐ No ______________________ ______________________

Like it? ☐ Yes ☐ No ______________________ ______________________

Product Name: ______________________ ______________________

Description: ______________________ ______________________

Problem? ☐ Yes ☐ No ______________________ ______________________

Like it? ☐ Yes ☐ No ______________________ ______________________

Product Name: ______________________ ______________________

Description: ______________________ ______________________

Problem? ☐ Yes ☐ No ______________________ ______________________

Like it? ☐ Yes ☐ No ______________________ ______________________

FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST• FAST

PERFORMANCE PLUS

See Wordstar™, dBASEII™, and other Z80 software really fly!

SLAVE PROCESSORS FOR 11.27 or 31.05 CROMIX

8MHZ Z80H 256K

TWO USERS PER SLAVE

Increase Performance Two to Fourteen Times

Much Faster Terminal I/O

$1095.00 includes software license

(Also ideal for fast process control applications)

NEW 32K STRUCTURED BASIC

Formatted input — High speed built-in Sort

Programs load up to FIVE TIMES FASTER

CROMIX (SBASIC.BIN)	$295
CP/M	$195
UPDATE CROMIX VERSION	$95*
UPDATE CDOS VERSION	$95*

* Must be holder of valid Cromemco License

RUN YOUR FAVORITE SBASIC PROGRAMS ON PC/MS-DOS SYSTEMS

SBASIC.PC	$295
SBASIC.PC RUNTIME	$100

SYSTEMS ATLANTA, INC., P.O. BOX 99, LEBANON, GA 30146 (404) 928-0240

Wordstar is a trademark of Micropro. dBASEII is a trademark of Ashton-Tate.

•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST• FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•

CROMEMCO COMPUTERS: DESIGNED TO MAKE UNIX SYSTEM V EVEN BETTER...

UNIX System V, the new standard in multiuser microcomputer operating systems, gives you high performance features along with the portability and flexibility of a standard.

Cromemco computers can make UNIX System V even better. Because our systems are designed with UNIX in mind. First of all, we offer UNIX System V with Berkeley enhancements. Then, our hardware uses advanced features like 64K of on-board cache memory and our high speed STDC controller to speed up disk operations – very important with UNIX.

More capability and expandability

We have a high-speed, 68000-based CPU that runs at 10 MHz, coupled with a memory manager that uses demand-paging and scatter loading to work *with* UNIX, not for it.

We provide room for expanding RAM to 16 megabytes – with error detection and correction – for running even the most sophisticated and advanced microcomputer programs. And the power to accommodate up to 16 users – all with plenty of memory.

But we give you even more.

A complete solution

We give you a choice in systems: the System 100 series, expandable up to 4 megabytes of RAM, and the System 300 series, expandable to 16 megabytes. A high speed 50 megabyte hard disk drive is standard on the systems. And you can expand the hard disk capacity up to 1200 megabytes using standard SMD drives. You can add floating point processing. High resolution graphics. Video digitizing and imaging. Communications through standard protocols. Mainframe interface.

And software support is here to meet your needs. We offer major programming languages, database management systems, communications software, including SNA architecture, X.25 protocol, and Ethernet; even a program to interface to an IBM PC if you need to. And, of course, access to the broad range of standard UNIX applications programs that is growing dramatically every day.

Easy to use.

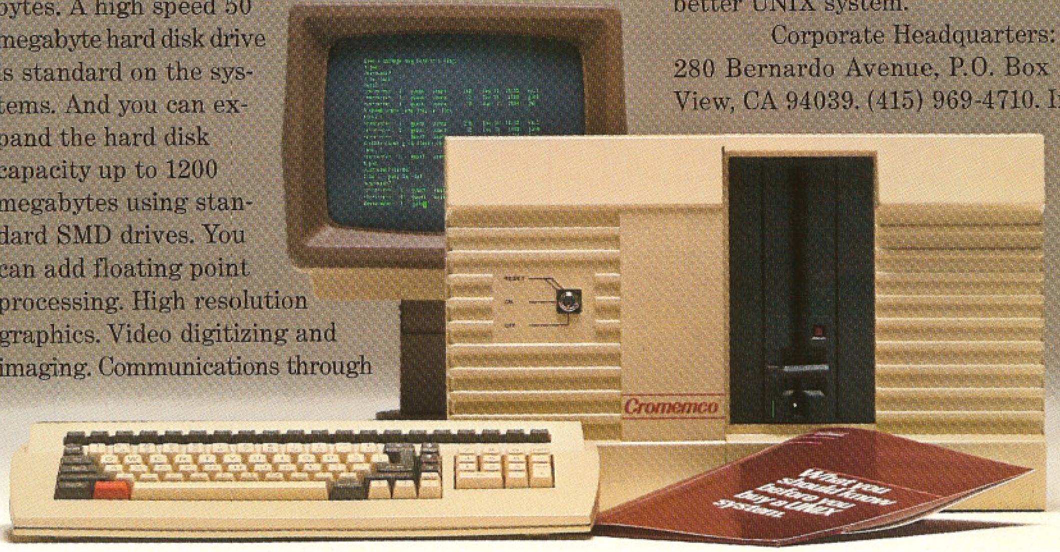
We also make our systems easier to use, because we install the operating system before we ship your computer. No complicated installation procedures. And the Berkeley enhancements give you the standard UNIX System V operating system, but with the added convenience of these widely acclaimed improvements.

Cromemco's System 100 and System 300 computers: designed to be the highest performance UNIX systems available anywhere.

Just call or visit one of our UNIX System V Official System Centers to see for yourself. They'll also give you a copy of our new publication, "What you should know before you buy a UNIX system." Or contact us directly.

We'll be glad to show you how to get a better UNIX system.

Corporate Headquarters: Cromemco, Inc., 280 Bernardo Avenue, P.O. Box 7400, Mountain View, CA 94039. (415) 969-4710. In Europe: Cromemco GmbH, 6236 Eschborn 1, Frankfurter Str. 33-35, P.O. 5267, Frankfurt Main, Germany.



UNIX is a trademark of Bell Laboratories.
IBM is a trademark of International Business Machines Corp.

Cromemco®